ICON Integration Adapter for SOM TA Investor PartyRole

Detailed Design Document

Version Draft | Jun 24, 2019

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Update summary | Updated by | Update date |
|
| 0.1 | Initial draft | Roshan Kumar Singh | 24/06/2019 |
| 0.2 |  |  |  |
|  |  |  |  |
|  |  |  |  |



Contents

[1. Introduction 5](#_Toc12532700)

[1.1 Reference 5](#_Toc12532701)

[2. Scope, Dependencies & Assumptions 7](#_Toc12532702)

[2.2 Scope Items 7](#_Toc12532705)

[2.3 Assumptions 8](#_Toc12532706)

[2.4 Scope Exclusions and Dependencies 8](#_Toc12532707)

[3. Overview of the requirement 8](#_Toc12532708)

[3.1 Requirements 8](#_Toc12532709)

[4. System Overview 9](#_Toc12532710)

[4.1 System Context 9](#_Toc12532714)

[4.2 Actors (w.r.t. System Context) 9](#_Toc12532715)

[4.3 High Level Integration Component View 9](#_Toc12532716)

[4.4 Functions and Component Model 9](#_Toc12532717)

[4.5 Error handling and TED 9](#_Toc12532718)

[4.6 Sequence diagram of events (IU to core system) 9](#_Toc12532719)

[5. ICON Integration Adapter High Level Architecture 11](#_Toc12532720)

[5.1 Technical Assumption 11](#_Toc12532725)

[5.2 Technical Architecture Overview Diagram 11](#_Toc12532726)

[5.3 Illustration of the Technology view 11](#_Toc12532727)

[5.4 Description of High-Level Components 11](#_Toc12532728)

[6. UI Design 12](#_Toc12532729)

[7. Architectural Decisions 12](#_Toc12532730)

[7.1 Architectural Decisions 12](#_Toc12532732)

[7.2 Audit Logging related Requirement Implementation Approach 12](#_Toc12532733)

[8. Component Model 13](#_Toc12532734)

[8.1 Reader Router Component/Class Diagram 13](#_Toc12532741)

[8.2 ICON Transformer component/Class diagram 13](#_Toc12532742)

[8.3 ICON Writer Processor class diagram 14](#_Toc12532743)

[8.4 Reference Data Service 14](#_Toc12532744)

[8.5 Cross Reference ID Mapping 15](#_Toc12532745)

[9. Business Entity Relationship 15](#_Toc12532746)

[10. Entity State Machines 15](#_Toc12532747)

[11. Event payload Structure 15](#_Toc12532748)

[12. Datastore Design Overview 15](#_Toc12532749)

[13. Detailed Design of Adapter 15](#_Toc12532750)

[13.1 ICON Reader Router service (adapter-partyrole-out-rdr) 15](#_Toc12532752)

[13.1.1 List of Major components 15](#_Toc12532753)

[13.1.2 Implementation Details 16](#_Toc12532754)

[13.1.3 Component Sequence Diagram (SOM TA Outbound Flow) 17](#_Toc12532755)

[13.1.4 Exception Events 17](#_Toc12532756)

[13.1.5 Configurations details (Application.yml/ConfigMap.yaml) 19](#_Toc12532757)

[13.2 ICON Transformer Service (Adapter-Icon-PartyRole-Out-MTF) 20](#_Toc12532758)

[13.2.1 List of Major Components 20](#_Toc12532759)

[13.2.2 Implementation Details 20](#_Toc12532760)

[13.2.3 Component Sequence Diagram 23](#_Toc12532761)

[13.2.4 Exception Events 23](#_Toc12532762)

[13.2.5 Configurations details (Application.yml/ConfigMap.yaml) 26](#_Toc12532763)

[13.3 ICON Writer Processor Service (Adapter-icon-partyrole-out -ewp) 26](#_Toc12532764)

[13.3.1 List of Major Components 26](#_Toc12532765)

[13.3.2 Implementation Details 27](#_Toc12532766)

[13.3.3 Component Sequence Diagram (Outbound Flow) 29](#_Toc12532767)

[13.3.4 Exception Events 29](#_Toc12532768)

[13.3.5 Configurations details (Application.yml/ConfigMap.yaml) 32](#_Toc12532769)

[13.4 ICON Cross Ref ID Processor (Adapter-FuCom-Out-XREF) 32](#_Toc12532770)

[13.4.1 List of Major components 32](#_Toc12532771)

[13.4.2 Implementation Details 33](#_Toc12532772)

[14. Source-To-Target Message Transformation 34](#_Toc12532773)

[14.1 Source to Target Mapping 34](#_Toc12532782)

[14.2 ICON API contract 35](#_Toc12532783)

[14.3 ICON API Definition (Swagger) 35](#_Toc12532784)

[15. Traceability & Testing Strategy 36](#_Toc12532785)

[15.1 Traceability (Business Use cases to Tech Stories) 36](#_Toc12532795)

[15.2 Testing Strategy 36](#_Toc12532796)

[15.3 Unit Testing 37](#_Toc12532797)

[16. Integration Adapter Repository List and Kafka Topics 37](#_Toc12532798)

[16.1 Kafka Topics List 37](#_Toc12532802)

[16.2 Bit Bucket Repository List 39](#_Toc12532803)

[17. Microservices Design 39](#_Toc12532804)

[17.1 HTTP Response Codes 39](#_Toc12532806)

[18. Security Architecture 40](#_Toc12532807)

[19. Deployment 40](#_Toc12532808)

[20. Glossary & Abbreviations 40](#_Toc12532809)

1. Introduction

This document covers detailed design for Icons integration adapter. Functional scope of the design document is to cover handling of *Investor PartyRole* event generated by upstream utility system Investor Utility (IU) and published to the downstarem core system ICON.

ICON Integration adapter is to allow integration between IU & ICON, is built using loosly coupled microservices components running independent of source (IU) and target (ICON) system. It uses Kafka technology as underlying asynchronous event based integration fabric to provide robust integration layer.

This is a running document, will be updated throughout the development & maintenace phases. Please refer latest verison of reference document referred through out this document.

### Reference

Following set of documents are refered at various sections in the detailed design. It is necessary to refer these document to understand full context in those relevant sections.

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Document Name | Description | Location |
|
| Ref-01 | xReference API Guide.pdf | ID Cross refererence guide | https://ipwc.ihost.com/us/ntrs/goto.do?detail=FI\_360428 |
| Ref-02 | FundUtility\_Detailed\_Design\_document\_vX.X.docx | Fund Utility detailed design | https://ipwc.ihost.com/us/ntrs/goto.do?detail=FO\_347092 |
| Ref-03 | Icon\_apis.zip | Swagger for ICON APIs | https://ipwc.ihost.com/us/ntrs/goto.do?detail=FI\_360064 |
| Ref-04 | Draft\_SIT\_Test\_Scenarios\_SIT\_Soft\_V01.xlsx | ICON Integration Testing Strategy and test cases | TBD |
| Ref-05 | Icon\_apis.zip | Contains field level details of API request/response attributes | https://ipwc.ihost.com/us/ntrs/goto.do?detail=FI\_360064 |
| Ref-6 | SOMTA out Bound Adapter Common Detailed Design\_v2.0\_WIP | Common detailed design for core system adapters | https://ipwc.ihost.com/us/ntrs/goto.do?detail=FI\_360431 |
| Ref-7 | Exception events from Adapters |  | https://ipwc.ihost.com/us/ntrs/goto.do?detail=FO\_359650 |

1. Scope, Dependencies & Assumptions

### Scope Items

Functional scope of the ICON adapter is to handle *Investor PartyRole* business event generated by source system Investor Utility (IU) wherein (ICON) acts as a consumer of the event. Adapter is to consume the event generated by upstream system (IU) and downstream to the core system (ICON). Any error or exceptional scenarios during will be handled to maintain consistency

Technical scope of the ICON adapter is to build a set of adapter services to read, transform and consume appropriate ICON APIs to create/update entities/sub-entities in core system. Scope also covers graceful handling of error or exceptional scenarios during the integration.

During any update IU will publish the entire Investor PartyRole Payload with different version number and PartyRole domain entity will have amend=true/false to indicate whether it’s create or update. Also, the domain entity will have a Technical Attribute called ‘previousVersion’ = < self-link of the previous version of the entity.

Sample section which will be available in domain entity jsons

{

"rel": "self",

"href": "http://partyroleservice.ksd.ntrs.com/v1/partyRole/fd9bcc04-e28a-4445-8777-dd196afc78a7-1",

"hreflang": null,

"media": null,

"title": null,

"type": null,

"deprecation": null

},

{

"rel": "previousVersion",

"href": "http://partyroleservice.ksd.ntrs.com/v1/partyRole/d0bffef2-b37e-47c9-a59b-4324ad532e45",

"hreflang": null,

"media": null,

"title": null,

"type": null,

"deprecation": null

}

Amend field in the json

"dthNtcSndByPtyRlIdTxt": null,

"lglEntity": null,

"amend": true,

Following tech stories are covered:

|  |  |  |  |
| --- | --- | --- | --- |
| Tech story (Jira ref) | Description | Mapped ICON API | Jira link |
|
| MIA-166 | Create Investor PartyRole profile in ICON once Investor Utility publishes the 'Create PartyRole' payload | ICON *Party*  API (POST) |  |
| MIA-166 | Handle event generated after IU updates exsiting Investor Party Role & publishes the 'Update PartyRole' *update* event | ICON *Party*  API (PUT) |  |
|  |  |  |  |

### Assumptions

None.

### Scope Exclusions and Dependencies

Other ICON inbounds Investor events (e.g. InvestmentAccount, AgentFundParticipation & PartyRoleRelationship) are not in scope of this document. ICON Outbound events are opting in the scope of this document.

This document compliments a Common detailed design for core system adapters [Ref-06] that detailed many common technical design aspects of core system adapters.

1. Overview of the requirement

### Requirements

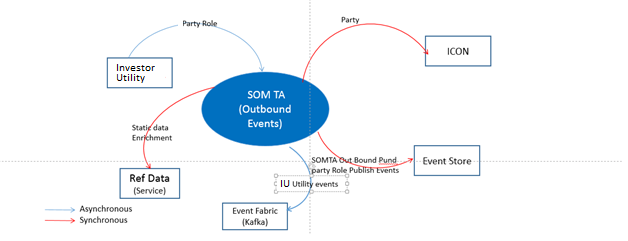
From the portal UI once the party role is submitted after rounds of processing (i.e. through Api or State Machine) it gets published and that event will come to Integration Adapters components. The first adapter component Readerouter will subscribe to the PartyRole publish event from the IU topic and then apply routing/filtering logic to determine which core engine (ICON) will be updated. Once this is done payload will be transformed inside Transformer as per the transformation rules and mapping of core engine / utility mapping and then Writer Process will be invoked. Writer Processor adapter component in turn will be invoking core engine apis to provision the PartyRole in the core engine and publish a json to its out topic which conforms to the Cross Ref Id mapping schema and Cross ref Id processor will subscribe to that topic and get the cross ref id CRUD operation json and will perform CRUD operation using the Cross ref ML Api.

Please refer below list of Investor Utility use cases which will be triggering the party role events which will be consumed by this Adapter component

* UC\_2\_New Client Setup - Investor Party Role creation
* IU23\_New Client Setup – Approval and if not eligible QCAml flow
* IU\_23 &IU\_24\_New Client Setup – Approval if QCAml
* IU\_24, 25, 26 \_New Client Setup - Approval if Manual Aml

1. System Overview

### System Context



### Actors (w.r.t. System Context)

Please refer to the section 3.1 Actors with respect to the system context of common design document [[Ref-06](#ref06)]

### High Level Integration Component View

Please refer to the section 3.3 High Level Integration Component View of common design [[Ref-06](#ref06)].

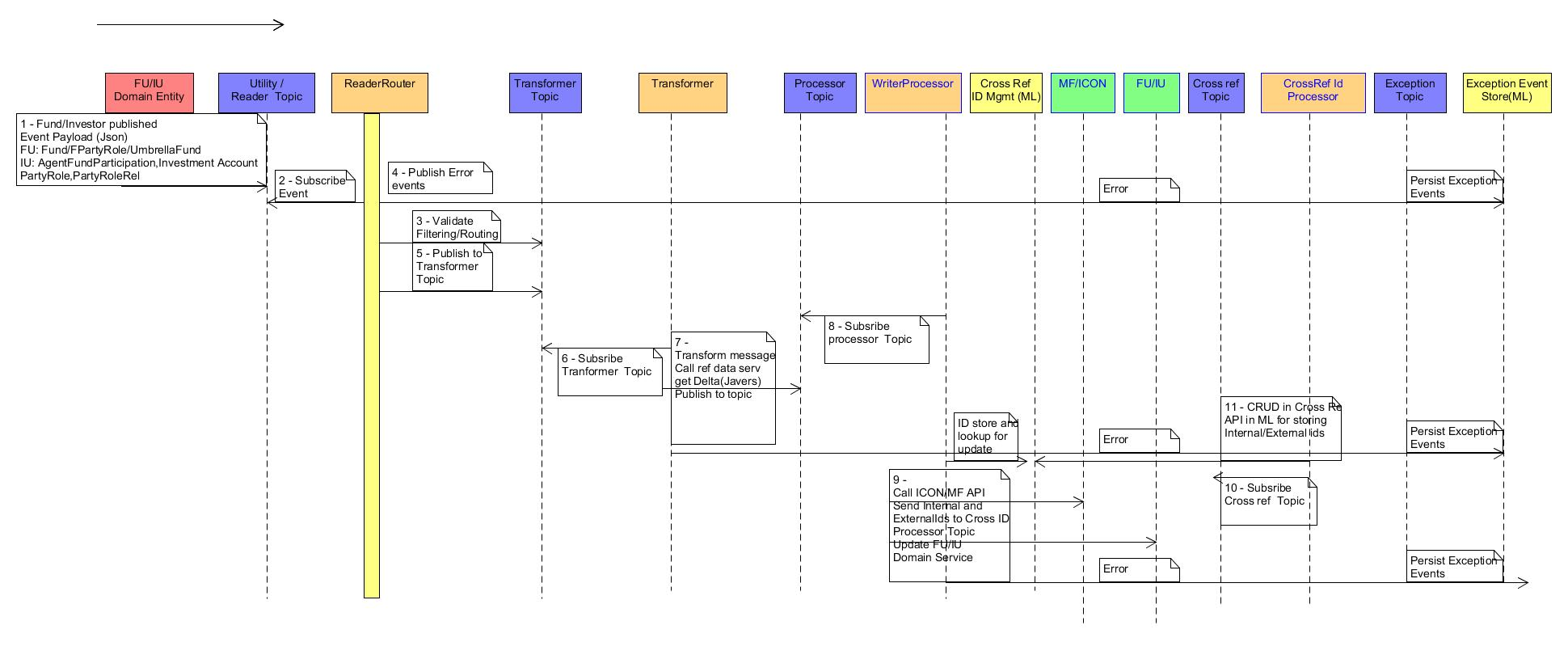
### Functions and Component Model

Please refer to the section 3.4 of High Level Integration Component View of common design [[Ref-06](#ref06)].

### Error handling and TED

Please refer to the section 3.5 of High Level Integration Component View of common design [[Ref-06](#ref06)].

### Sequence diagram of events (IU to core system)



1. ICON Integration Adapter High Level Architecture

### Technical Assumption

Please refer to the section 4.1 Technical Assumption common design [[Ref-06](#ref06)].

### Technical Architecture Overview Diagram

Please refer to the section 4.2 Technical Assumption common design [[Ref-06](#ref06)].

### Illustration of the Technology view

Please refer to the section 4.2 Technical Assumption common design [[Ref-06](#ref06)].

### Description of High-Level Components

Adapter Framework Components supports various responsibilities such as File/Event Reader, Event Processing, Event Publisher and Source Transformation. Please refer to Adapter Framework Design [Ref-x] for the detail.

ICON Integration adapter microservices will leverage these adapter framework components for these common functionalities. Below are the high-level ICON integration adapter microservice components.

**ICON Message Reader and Router:** This component will be responsible for reading published messages from outbound Kafka topic. This component will leverage adapter framework Source Reader component for reading messages. Then it will determine the destination (Multifonds/ICON) based on business logics (values of few fields e.g. TA Onshore/ Offshore or LoB Indicator in the domain entity).

* Reader Router Adapter Microservice will listen to Investor Party Role topics for the published events from the Investor utilities (IU). IU will publish Investor PartyRole, PartyRoleRelationship, InvestmentAccount and AggentFundParticipation on four different topics.
* Router function will first parse for the technical fields (TA Onshore/ Offshore or LoB Indicator) published by IU in the event payload
* The technical field for determining the routing logic will be available in the EventContext
* Based on the Technical attributes sent by Investor Utility, router function will determine destination core system (MF or ICON). Then it will pass the message to respective transformer topic.
* This reader component will be common for MF/ICON.
* Spring boot integration stream Pollable interface will be implemented for listening to multiple topics from different utilities and published to adapter specific topics based on the routing logic for the consumption of domain specific integration adapters**.**

**ICON Event Transformer:** This component is a generic publisher spring cloud stream “sink” type adapter does following:

* Reads the messages with enriched header information from Event Processor from specific Kafka topic
* Map source to target data and does field transformation.
* Determine delta for update (in case of ICON core target system)
* Enrich the messages by invoking reference data and typecode category services
* Publish the transformed message to a Topic
* Store Base Exception in Domain Adapter Event
* Publish exception to the exception topic and store exception events in Marklogic (ML) DB
* Clear exception and emit Exception update event

**ICON Writer Processor (Publisher):**

This adapter service is responsible for invoking ICON APIs for create/Update party in core system.

* This component calls REST APIs of ID Cross Reference Management system. This is to retrieving ID mapping from the system
* Then calls appropriate ICON APIs to create/update domain entities in ICON core system. ICON internal IDs are used in case of update operations.
* This component has retry logic (e.g. if ICON system does not respond or system is not reachable due to intermittent network issue)
* Generate event encapsulating status (API response) from the core and publish to Fund Utility

**Cross Reference ID Processor**

This adapter service will be responsible for CRUD operation in the Cross Reference ID store in ML database. Key function of this service is to update Internal and external ID mapping in the target system.

1. UI Design

Not Applicable for the Integration adapter

1. Architectural Decisions

### Architectural Decisions

Please refer to the section 5.1 Architectural Decisions of the common design [[Ref-06](#ref06)].

### Audit Logging related Requirement Implementation Approach

Please refer to the section 5.1 Architectural Decisions of the common design [[Ref-06](#ref06)].

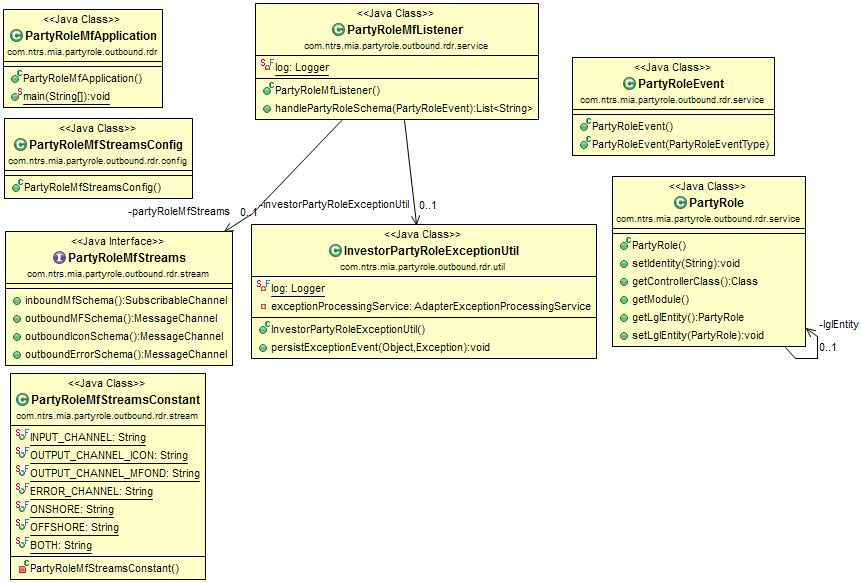
1. Component Model

Following section covers component models and class diagram for outbound integration adapter.



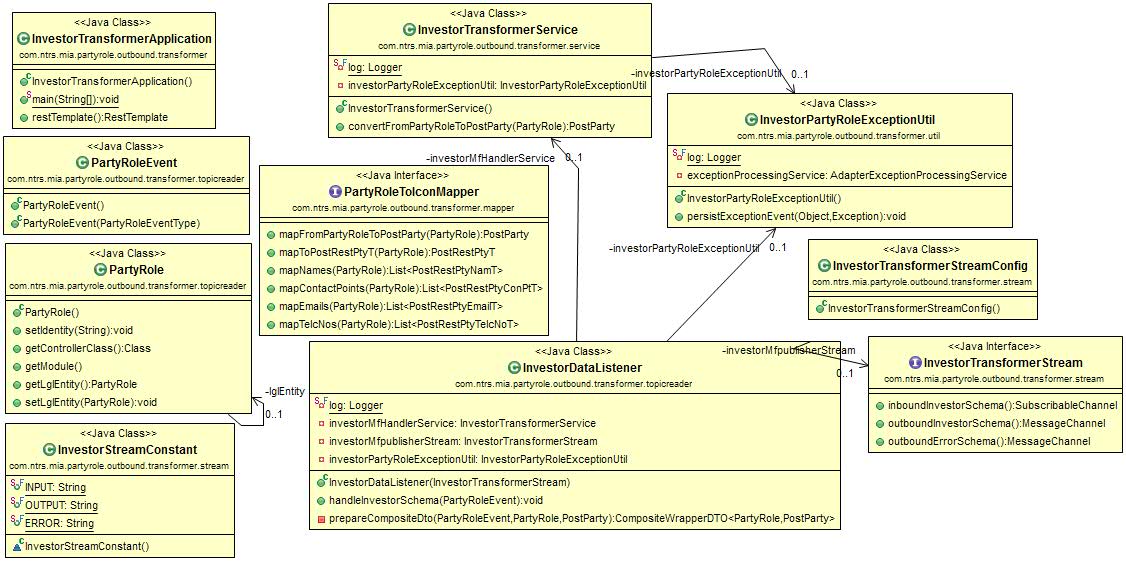
### Reader Router Component/Class Diagram

Please refer ….



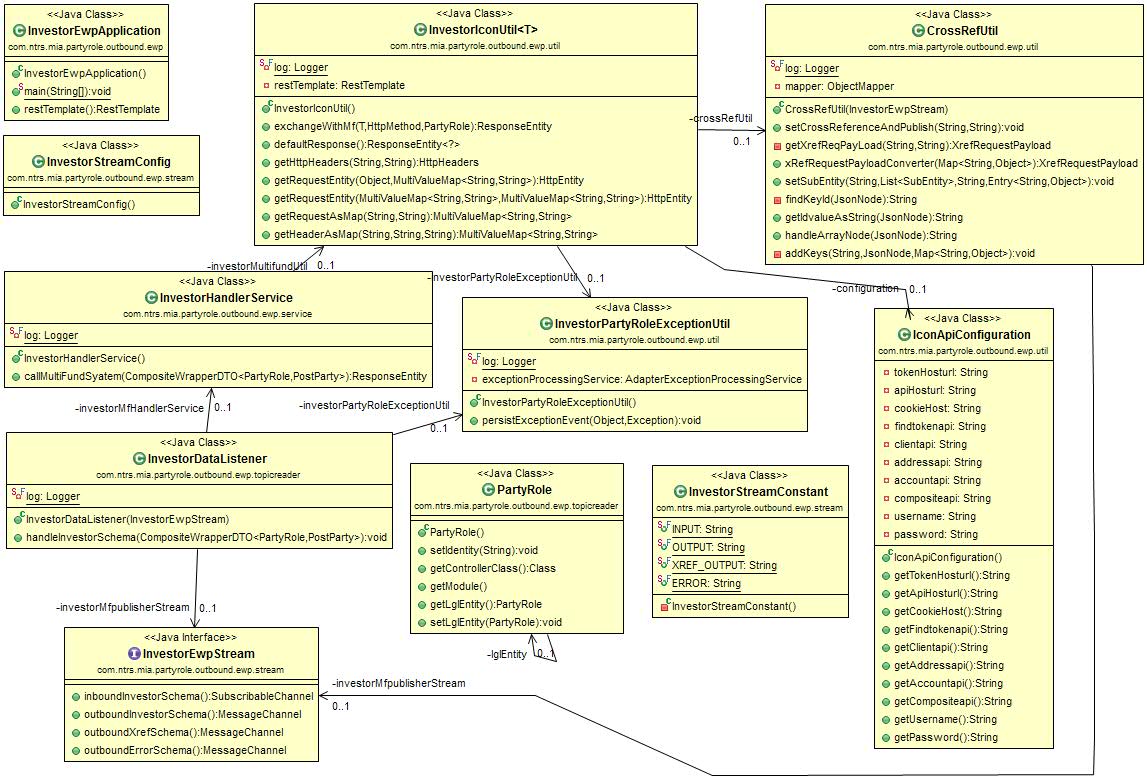
### ICON Transformer component/Class diagram

Following figure shows…



### ICON Writer Processor class diagram

Following figure shows…



### Reference Data Service

Please refer common design (REF-5) for the details of the Api and its component diagram

Please refer below table for the reference data mapping needed for this integration adapter

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Source Field(Utility PDM) | TypeCategoryName for Ref Data Api | TypeCodeValue from Ref data api | Target Api Json field | Any Transformation |
| 1 | CountryCode | CntryCode | US | ContactPoints.address | No |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

### Cross Reference ID Mapping

Cross reference Id mapping for Investor Party Role json and ICON Apis for creating party role

|  |  |  |  |
| --- | --- | --- | --- |
| # | Internal ID(Utility PDM) | External ID(Core Engine) | Mapping rule |
|
|  | PartyRole.uBERPtyID | PartyRole.reference.internalid | One to one |
|  |  |  |  |
|  |  |  |  |

Sub entity Cross Ref id mapping will be done after mapping exercise is complete

1. Business Entity Relationship

Please refer to the Fund LLD for details about Investor party role domain entity. Integration adapters don’t generate any domain entity or events

1. Entity State Machines

Integration adapters don’t generate any state change event as they are stateless.

1. Event payload Structure

The events are generated by Investor Utility and integration adapters are consumers for the same and refer to the Investor LLD for event payload structure

1. Datastore Design Overview

Adapter component don’t store any events in ML apart from exception events neither maintain any PDM. Hence it’s not applicable for the integration adapter

1. Detailed Design of Adapter

### ICON Reader Router service ([adapter-partyrole-out-rdr](http://adapter-fund-out-rdr/))

Following …

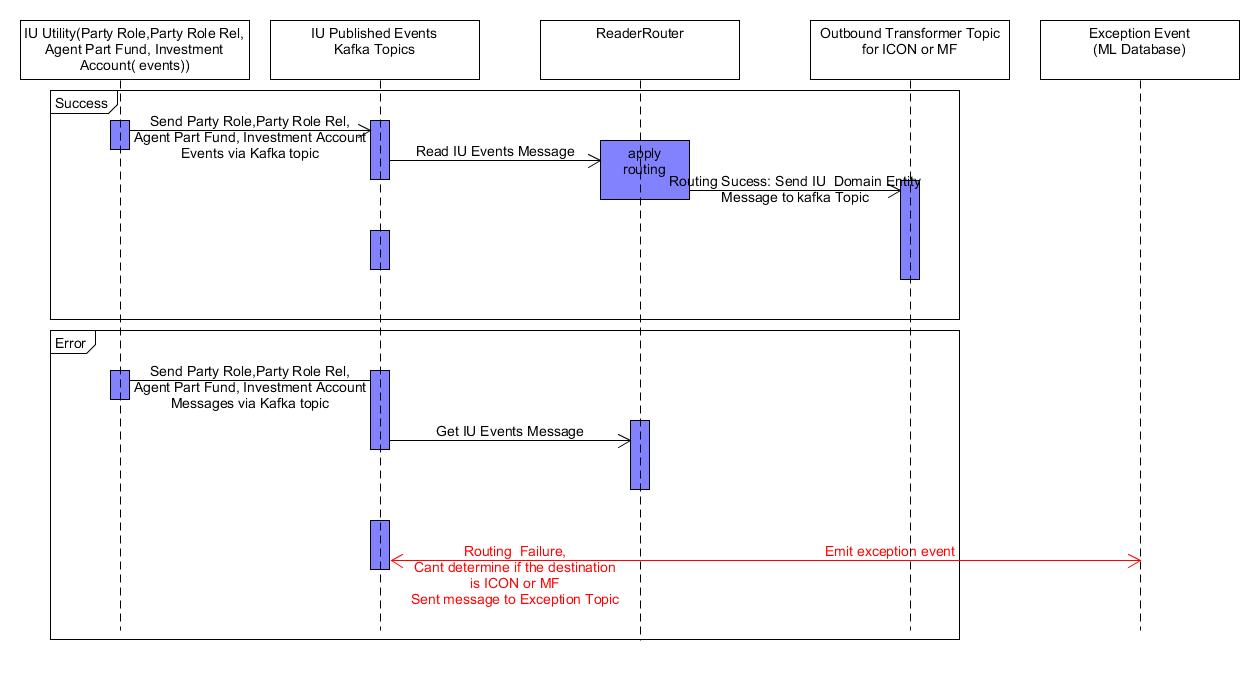
### List of Major components

| Package name | Method name | Input | Output | URL pattern |
| --- | --- | --- | --- | --- |
|
| com.ntrs.mia.partyrole.outbound.rdr.service.PartyRoleMfListener |  | (@payload <<Entity>> schema) <<Entity>> e.g. PartyRoleEvent, | PartyRole Domain entity | NA |
|  |  |  |  |  |

### Implementation Details

|  |  |  |
| --- | --- | --- |
|
|  | Business Logic | Input: PartyRole Publish event payload(json)  Output: PartyRole Domain Entity(json)  Please refer to the section 5.7 Fund and Investor Event Processing in Reader Router [[Ref-06](#ref06)] for details about the event processing  Reader service will read the published ICON Investor messages from the kafka topic by using spring integration stream.  Implement routing function(Refer section 11.3)  Reader component will be using spring boot annotation @ComponentScan to invoke adapter framework FileSource Reader component to read the json payload from the kafka topic.  Once the routing (Multifonds or ICON) is determined using the routing logic for FundPartyRole Entity – the payload will be written into another kafka topic using spring integration stream component. |
|  | Spring Boot Admin | Add dependencies for Spring Boot admin in POM  Config Map changes   * spring.boot.admin.client.url: [https://nt-spring-admin-server.ksd.ntrs.com](https://nt-spring-admin-server.ksd.ntrs.com/) * management.endpoints.web.exposure.include: "\*" * spring.boot.admin.client.instance.name: [adapter-fundprole-out-rdr](http://adapter-fund-out-rdr/) * spring.boot.admin.client.instance.service-url: [http://](http://adapter-fund-out-rdr/) [adapter-partyrole-out-rdr](http://adapter-fund-out-rdr/) [This should be K8 Service name] |
|  | Hystrix | Not applicable |
|  | Producer Side Message Compression | Below config Map properties added to the Config Map  spring.cloud.stream.kafka.bindings.<Topic Name>.producer.compressionType : gzip |
|  | Transaction handling | There is no requirement for business Transaction handling in this adapter. |
|  | Exactly Once Delivery | Would be done using consumer groups. Configure Consumer Group, Minimum partition, instance-count, instance-index as shown in following application.yml file as red box  Pseudo configuration |
|  | TNT Crypt Encryption/Decryption | Channel encryption (Encrypt while putting message to Kafka and Decrypt while receiving message from Kafka)  Not applicable for Fund Utility as none of the fields for Fund described as PI Sensitive |
|  | Audit event | Not applicable for Integration adapter |
|  | K8 Service URL | [http://](http://adapter-fund-out-rdr/) [adapter-partyrole-out-rdr](http://adapter-fund-out-rdr/) |
|  | Common Library | AdapterExceptionEvent-Handler for handling exception events  AdapterCommon-Out |
|  |  |  |

### Component Sequence Diagram (SOM TA Outbound Flow)



### Exception Events

Please refer below for the exception event emitted by this component.

| # | Adapter | Exception | Exception Type | Reason | Replay | Error code | Error Name | Error description |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Routing logic failed due to incorrect data from Utilities | Technical Exception |  | No Can’t be replayed from TED and manually handled and event need to be reinitiated from Source | INTG\_Non\_Replayable\_01 | Routing logic failed | Routing logic failed for Domain entity = [Ex: Fund,Investor etc] and for field =[Fully qualified field name]] |
|  |  |  |  |  |  |  |  |  |

Please refer below sample exception event

|  |
| --- |
| { "eventId": "22ef70fb-73e1-44e8-b023-069274c4f511", **-->Auto generated in the adapter would be used for lookup when replay triggered** "eventTimestamp": 1559545545140,  "timeZone": "UTC",  "payload": { "exceptionInfo": { "coreEventPayload": null**, --> Not applicable here** "name": "Adapter Exception",  "description": " Routing logic failed for Domain entity ",  "exceptionClass": "ADAPTER",  "type": null,  "status": "NEW",  "errors": [ { "errorCode": "INTG\_Non\_Replayable\_01", **--> will be used for storing error code emitted by the adapter** "errorMessage": " Routing logic failed for Domain entity = FundpartyRole and for field =PartyRole.taOnshoreOffshoreIndicator ", **will be used for storing error message emitted by the adapter** "errorType": "Replayable Error" } ],  "exceptionContext": { "UUID": "4424424242", **🡪 Domain Entity Uber id** "transactionFlowType": "SUBS" },  "domainId": "4424424242", **🡪 Domain Entity Uber id** "exceptionID": "53ab4816-d7b7-4fd8-94c1-a88eb27e9757",  "links": [ ],  "PartyRole": { **--> Source Event Payload** "taOnshoreOffshoreIndicator": "onshore",  "UBERFundId": "10",  "FundType": "O",  "IsAmmendable": false,  "SingleSwingPricingParametersArray": [ { "SingleSwingPricingParametersOperationType": "",  "SingleSwingPricingParametersPricingThresholdMethodType": "" } --- --- },  "eventClass": "ADAPTER\_EXCEPTION\_EVENT",  "domainName": "FundPartyRole",  "eventHeader": { "previousEventId": null,  "sourceEventId": null,  "currentEventId": null,  "sourcePublisherId": null,  "sourceEventTimestamp": null,  "publisherId": null,  "mode": "MANUAL" },  "eventContext": { },  "eventName": " PartyRole\_PUBLISH",  "replayable": true,  "links": [ { "rel": "replay", --> **Replay Link** "href": "[http://localhost:8080/v1/replay/eventId=22ef70fb-73e1-44e8-b023-069274c4f511&eventName= PartyRole\_PUBLISH /accept](http://localhost:8080/v1/replay/eventId=22ef70fb-73e1-44e8-b023-069274c4f511&eventName=%20PartyRole_PUBLISH%20/accept)",  "hreflang": null,  "media": null,  "title": null,  "type": null,  "deprecation": null } ] } |

### 13.1.5 Configurations details (Application.yml/ConfigMap.yaml)

|  |  |  |
| --- | --- | --- |
| Sl # | Configuration Property | Sample values |
|  | spring.cloud.stream.bindings.partyrole-in.destination | nt\_mia\_party\_role\_mf\_out\_reader\_event\_dev1 |
|  | spring.cloud.stream.bindings.partyrole-in.content-type | application/json |
|  | spring.cloud.stream.bindings.partyroleicon-out | nt\_mia\_party\_role\_icon\_out\_transformer\_event\_dev |
|  | spring.cloud.stream.bindings.partyroleicon-out.content-type | application/json |
|  | spring.cloud.stream.bindings.partyrolemf -out.destination | nt\_mia\_party\_role\_mf\_out\_transformer\_event\_dev |
|  | spring.cloud.stream.bindings. partyrolemf -out.content-type | application/json |
|  | spring.cloud.stream.bindings.error-out.destination | nt-mia-iu-mf-exception |
|  | spring.cloud.stream.bindings.error-out.content-type | application/json |
|  | spring.cloud.stream.kafka.binder.brokers | ut11241.ntrs.com:9092 |
|  | spring.cloud.stream.kafka.binder.auto-create-topics | "false" |
|  | spring.jackson.mapper.accept\_case\_insensitive\_properties | "true" |

### ICON Transformer Service (Adapter-Icon-PartyRole-Out-MTF)

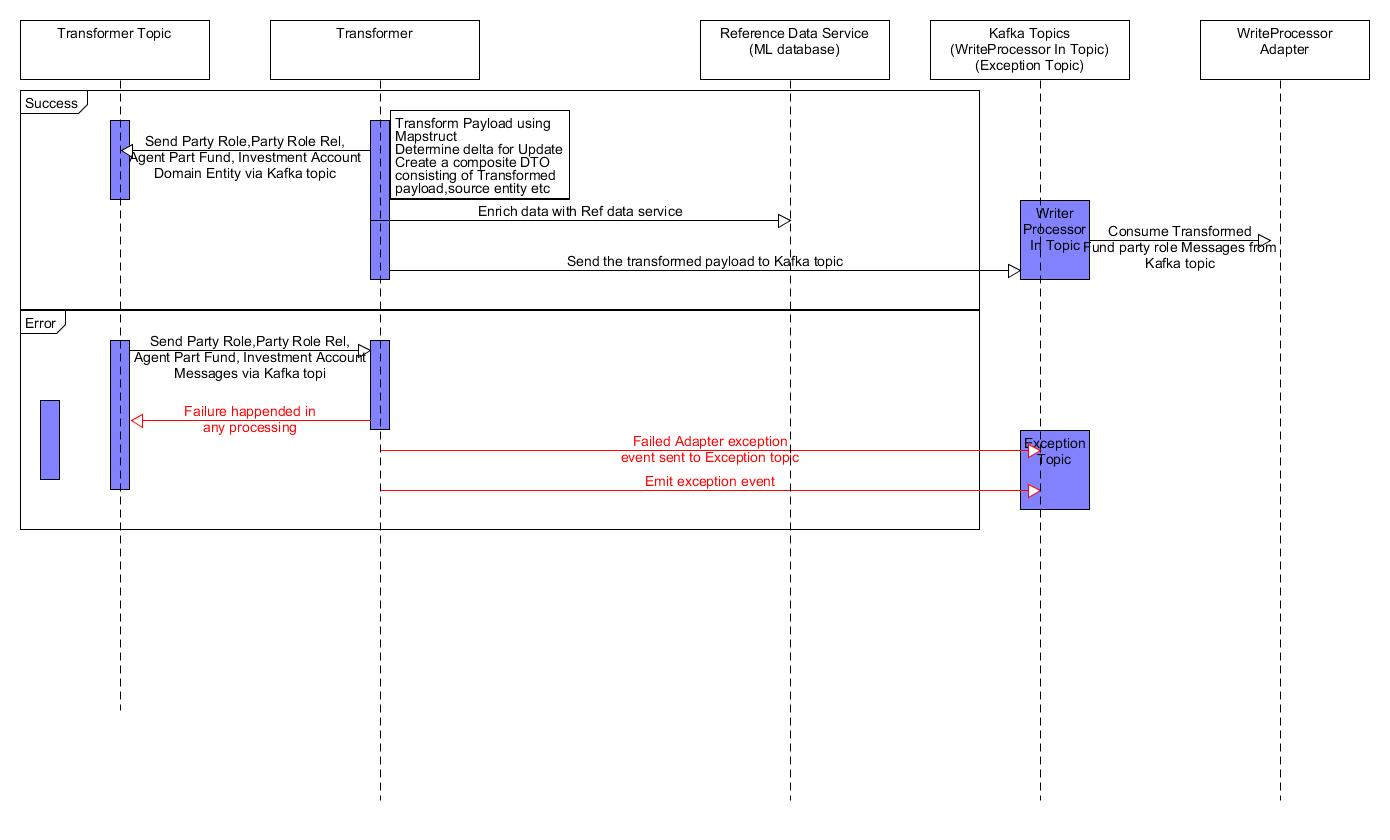
### List of Major Components

| Package name | Method name | Input | Output | URL pattern |
| --- | --- | --- | --- | --- |
|
| com.ntrs.mia.partyrole.outbound.transformer.topicreader. InvestorDataListener |  | (@payload <<Entity>> schema) <<Entity>> e.g. PartyRole, | compositeWrapperDTO JSON String | NA |
|  |  |  |  |  |

### Implementation Details

|  |  |  |
| --- | --- | --- |
|
|  | Business Logic | Input: Domain Entity <<Entity>> e.g. PartyRole  Output: compositeWrapperDTO - converted to JSON String . Here target will contain the transformed json payload needed for calling ICON api. Please refer below picture for the same as you can see masterfund     1. Transformer will be using java mapstruct source to target mapper for the transformation.AdapterSourceTransformer will not be used as we are ony using mapstruct here 2. Transformation service will invoke adaptor framework mapstruct wrapper factory to pass source and target object and the mapping template during runtime 3. Framework will apply this mapping to the target object stream and create a composite DTO described above and inject hateoas link for Entity and source entity . 4. This will also enrich the transformed payload using reference data by calling Reference data apis 5. For update events (determined by Amend=true attribute in partyrole payload) - this will use the javers to determine the delta and will start updating the composite dto to form the ICON update payload 6. Finally convert the composite dto to a json string and published to the kafka topic. 7. Implement Hystrix while calling reference data api |
|  | Spring Boot Admin | Add dependencies for Spring Boot admin in POM  Config Map changes   * spring.boot.admin.client.url: [https://nt-spring-admin-server.ksd.ntrs.com](https://nt-spring-admin-server.ksd.ntrs.com/) * management.endpoints.web.exposure.include: "\*" * spring.boot.admin.client.instance.name: Adapter-icon-partyrole-out-mtf spring.boot.admin.client.instance.service-url: [http://](http://adapter-fund-out-rdr/) Adapter-icon-partyrole-out-mtf [This should be K8 Service name] |
|  | Hystrix | Implement Hystrix while calling reference data api  In the Spring Boot Application add the below annotation(Pseudo Code)  @EnableCircuitBreaker  In the method where we are invoking apis add the below annotation  @HystrixCommand(fallbackMethod = "referenceDataApiFallback")  Add fallback method and apply fallback logic  public void referenceDataApiFallback(PostParty postparty, Throwable e) {  log.info("referenceDataApiFallback#start: ");  } |
|  | Producer Side Message Compression | Below config Map properties added to the Config Map  spring.cloud.stream.kafka.bindings.<Topic Name>.producer.compressionType : gzip |
|  | Transaction handling | There is no requirement for business Transaction handling in this adapter. |
|  | Exactly Once Delivery | Would be done using consumer groups.. Configure Consumer Group, Minimum partition, instance-count, instance-index as shown in following application.yml file as red box  Pseudo configuration |
|  | TNT Crypt Encryption/Decryption | Channel encryption (Encrypt while putting message to Kafka and Decrypt while receiving message from Kafka)  Not applicable for Fund Utility as none of the fields for Fund described as PI Sensitive |
|  | Audit event | Not applicable for Integration adapter |
|  | K8 Service URL | http://Adapter-icon-partyrole-out-mtf |
|  | Common Library | AdapterExceptionEvent-handler  AdapterCommon-Out  PartyRole-Domain |
|  |  |  |

### Component Sequence Diagram



### Exception Events

Please refer below for the exception event emitted by this component

| # | Adapter | Exception | Exception Type | Reason | Replay | Error code | Error Name | Error description |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Transformer(SOMTA IN/Out Bound) |  |  |  |  |  |  |  |
|  |  | All System exception (Network down, Kafka down, ML Instance Down etc) | System Exception | ML Database Down,Kafka down and Network Down | Auto Replay | NA | NA |  |
|  |  |  | Reference data look up failed | Reference data look up failed | Yes can be replayed when data is fixed in ML using feeder file | INTG\_Replayable\_02 | Reference data look up failed | Reference data lookup failed for typecode=[] |
|  |  | Numberformatexception,nullpointerexception,dateformatexception etc | code Related exception | Tranformation logic failed due to coding error in parsing data - programatic error | Yes can be replayed when data or code is fixed | INTG\_Replayable\_03 | Tranformation logic failed due to coding error in parsing data - or wrong data | Tranformation logic failed in the field=[Fully qualified field name] |
|  |  | Field Validation error | Data related exception from source | Data validation failed for a particular field ( Missing Data element or Data validation fails) | Replay not possiblefrom TED but can not be replayed using replayed link but data need to be corrected from source and event need to be triggered again. | INTG\_Non\_Replayable\_10 | Tranformation logic failed due to missing field | Tranformation logic failed due to missing field = [Fully qualified field name] |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

Please refer below sample exception event

|  |
| --- |
| { "eventId": "22ef70fb-73e1-44e8-b023-069274c4f511", **-->Auto generated in the adapter would be used for lookup when replay triggered** "eventTimestamp": 1559545545140,  "timeZone": "UTC",  "payload": { "exceptionInfo": { "coreEventPayload": null**, --> Not applicable here** "name": "Adapter Exception",  "description": " Reference data look up failed ",  "exceptionClass": "ADAPTER",  "type": null,  "status": "NEW",  "errors": [ { "errorCode": " INTG\_Replayable\_02", **--> will be used for storing error code emitted by the adapter** "errorMessage": " Reference data lookup failed for typecode= CntryCode", **will be used for storing error message emitted by the adapter** "errorType": "Replayable Error" } ],  "exceptionContext": { "UUID": "4424424242", **🡪 Domain Entity Uber id** "transactionFlowType": "SUBS" },  "domainId": "4424424242", **🡪 Domain Entity Uber id** "exceptionID": "53ab4816-d7b7-4fd8-94c1-a88eb27e9757",  "links": [ ],  "PartyRole": { **--> Source Event Payload** "taOnshoreOffshoreIndicator": "onshore",  "UBERFundId": "10",  "Type": "O",  "IsAmmendable": false,  "SingleSwingPricingParametersArray": [ { "SingleSwingPricingParametersOperationType": "",  "SingleSwingPricingParametersPricingThresholdMethodType": "" } --- --- },  "eventClass": "ADAPTER\_EXCEPTION\_EVENT",  "domainName": "PartyRole",  "eventHeader": { "previousEventId": null,  "sourceEventId": null,  "currentEventId": null,  "sourcePublisherId": null,  "sourceEventTimestamp": null,  "publisherId": null,  "mode": "MANUAL" },  "eventContext": { },  "eventName": " PartyRole\_PUBLISH",  "replayable": true,  "links": [ { "rel": "replay", --> **Replay Link** "href": "[http://localhost:8080/v1/replay/eventId=22ef70fb-73e1-44e8-b023-069274c4f511&eventName= FundPartyRole\_PUBLISH /accept](http://localhost:8080/v1/replay/eventId=22ef70fb-73e1-44e8-b023-069274c4f511&eventName=ICONTrade/accept)",  "hreflang": null,  "media": null,  "title": null,  "type": null,  "deprecation": null } ] } |

### 13.2.5 Configurations details (Application.yml/ConfigMap.yaml)

|  |  |  |
| --- | --- | --- |
| Sl # | Configuration Property | Sample values |
|  | spring.cloud.stream.bindings.invSchema-in.destination | nt\_mia\_party\_role\_icon\_out\_transformer\_event\_dev |
|  | spring.cloud.stream.bindings. invSchema-in .content-type | application/json |
|  | spring.cloud.stream.bindings.invSchema-out .destination | nt\_mia\_party\_role\_icon\_out\_eventwriter\_event\_dev |
|  | spring.cloud.stream.bindings.invSchema-out .content-type | application/json |
|  | spring.cloud.stream.bindings.error-out .destination | nt-mia-iu-mf-exception |
|  | spring.cloud.stream.bindings.exception-output.content-type | application/json |
|  | spring.cloud.stream.kafka.binder.brokers | ut11241.ntrs.com:9092 |
|  | spring.cloud.stream.kafka.binder.auto-create-topics | "false" |
|  | source-transformer.mapperType | mapstruct |
|  | source-transformer.sourceFormat | json |
|  | spring.jackson.mapper.accept\_case\_insensitive\_properties | "true" |

### ICON Writer Processor Service (Adapter-icon-partyrole-out -ewp)

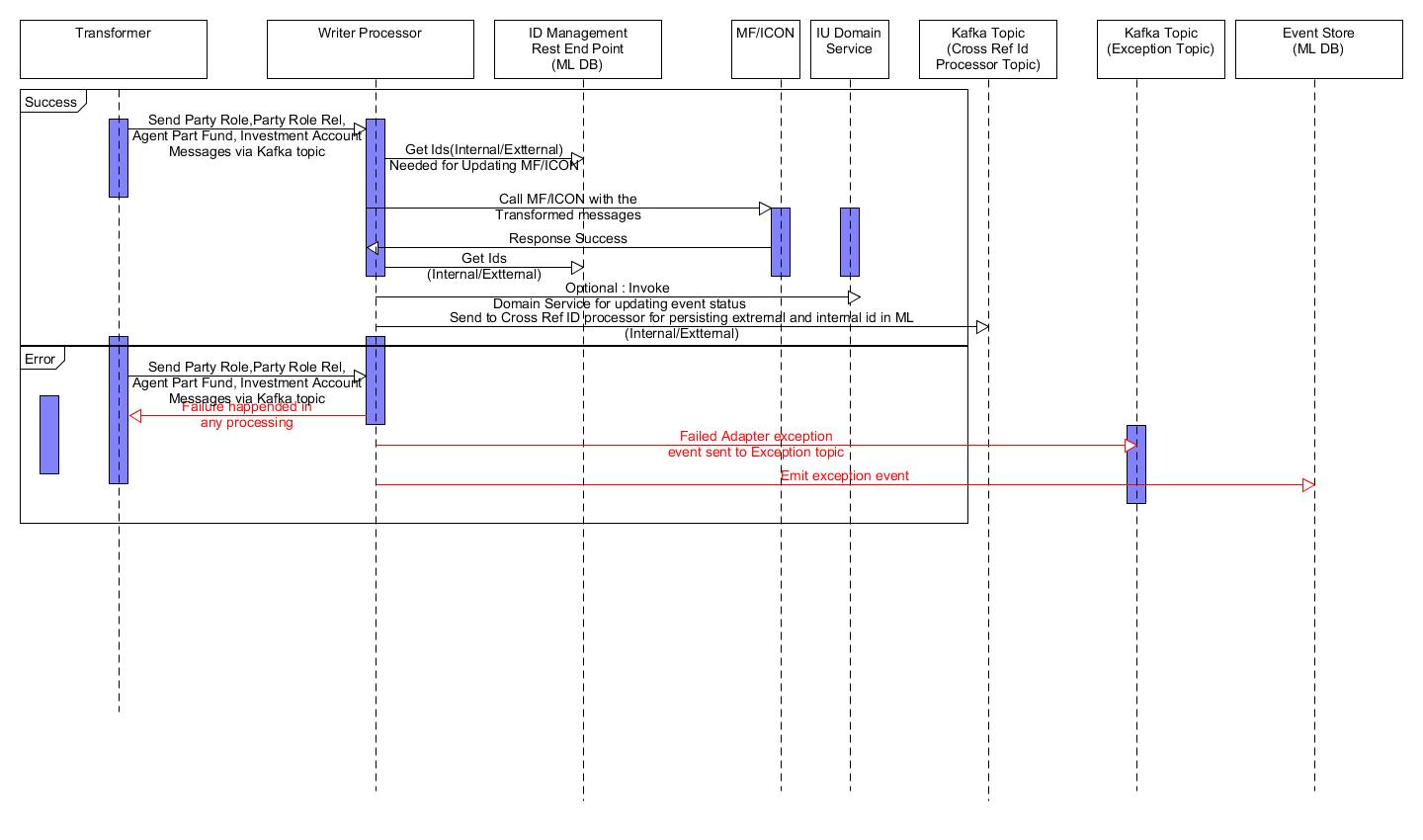
### List of Major Components

| Package name | Method name | Input | Output | URL pattern |
| --- | --- | --- | --- | --- |
|
| com.ntrs.mia.partyrole.outbound.ewp.topicreader | handleInvestorSchema | compositeWrapperDTO - JSON String | JSON String – which will be input for Cross Ref Api create/update action. | NA |

### Implementation Details

|  |  |  |
| --- | --- | --- |
|
|  | Business Logic | Input: compositeWrapperDTO - converted to JSON String . Here target will contain the transformed json payload needed for calling ICON api.  Output: JSON String – which will be input for Cross Ref Api create/update action   1. Write processor will parse the composite DTO and extract target api json string 2. WriterProcessor will use @RestTemplate to invoke ICON rest endpoints for provisioning of party role messages into ICON core engine. 3. After calling the ICON Apis - this will perform the Id mapping using the response from ICON apis and source event payload. To find out the source pdm input id (Ex: uBERPtyId using the source entity available in the composite dto) 4. Then this will form the json string for CRUD api call of Cross Reference Id management service and publish the json payload with action(create/update/delete) in the KAFKA topic so that Cross ref ID processor pick it up 5. Retry (retry 3 times) will be implemented in WriterProcessor and @Retryable ad @Backoff annotation will be used for the same. 6. In case of update – this will call the id management service to get the external ids for the core engine and update the transformed request corresponding ids of partyrole and then call the ICON apis.. 7. ICON will return the entire payload in the response and same Cross Ref Id management logic which is used for create will be leveraged to update the Cross ref Id management service. 8. Finally invoke PartyRole api to indicate that the partyRole is provisioned correctly in ICON 9. Implement Hystrix while calling ICON api and cross ref api |
|  | Spring Boot Admin | Add dependencies for Spring Boot admin in POM  Config Map changes   * spring.boot.admin.client.url: [https://nt-spring-admin-server.ksd.ntrs.com](https://nt-spring-admin-server.ksd.ntrs.com/) * management.endpoints.web.exposure.include: "\*" * spring.boot.admin.client.instance.name Adapter-icon-partyrole-out-ewp spring.boot.admin.client.instance.service-url: [http://](http://adapter-fund-out-rdr/) Adapter-icon-partyrole-out -ewp [This should be K8 Service name] |
|  | Hystrix | Implement Hystrix while calling reference data api  In the Spring Boot Application add the below annotation (pseudo Code)  @EnableCircuitBreaker  In the method where we are invoking apis add the below annotation  @HystrixCommand(fallbackMethod = "iconCreatePRoleFallBack")  Add fallback method and apply fallback logic  public void iconCreatePRoleFallBack (PostParty postparty, Throwable e) {  log.info(iconfCreatePRoleFallBack #start: ");  } |
|  | Producer Side Message Compression | Below config Map properties added to the Config Map  spring.cloud.stream.kafka.bindings.<Topic Name>.producer.compressionType : gzip |
|  | Transaction handling | There is no requirement for business Transaction handling in this adapter. |
|  | Exactly Once Delivery | Would be done using consumer groups.. Configure Consumer Group, Minimum partition, instance-count, instance-index as shown in following application.yml file as red box  Pseudo configuration |
|  | TNT Crypt Encryption/Decryption | Channel encryption (Encrypt while putting message to Kafka and Decrypt while receiving message from Kafka)  Not applicable for Fund Utility as none of the fields for Fund described as PI Sensitive |
|  | Audit event | Not applicable for Integration adapter |
|  | K8 Service URL | [http://](http://adapter-fund-out-rdr/) Adapter-icon-partyrole-out -ewp |
|  | Common Library | AdapterExceptionEvent-handler  AdapterCommon-Out – Pojos for ICON |
|  |  |  |

### Component Sequence Diagram (Outbound Flow)



### Exception Events

Please refer below for the exception event emitted by this component. Cross reference Id mapping for Party Role json and ICON Apis for creating fund (TBD – Waiting for mapping and rule)

| # | Adapter | Exception | Exception Type | Reason | Replay | Error code | Error Name | Error description |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | WriterProcessor(SOMTA Outbound) |  |  |  |  |  |  |  |
|  |  | jsonparseexception,nulpointerexception,numberformatexception. | Technical Exception | Parsing of the response payload failed - Can happen due to coding issues in publisher | Yes can be replayed when code is fixed | INTG\_Replayable\_04 | Parsing of the response payload failed | Parsing payload for [Ex : ICON or MF] failed for field [Fully qualified field name] |
|  |  | All System exception (Network down, Kafka down, ML Instance Down etc) | System Exception | ML Database Down,Kafka down and Network Down | Auto Replay | NA | NA |  |
|  |  | ML DB Cross Ref ID lookup failed | Technical Exception | Internal or extarnal ids missing in Cross Ref schema | Yes can be replayed after manually fixing the data using feeder file | INTG\_Replayable\_05 | ML cross reference Id look up failed | Lookup initiated for fieldName=[Qualified field name] and value= . Cross ref Id for Domain entity =[Ex: Fund/Investor ] and for field =[Fully Qualified field name] and Core entity Field=[] missing in ML DB. |
|  |  | IOException or others | Technical Exception | Domain entity gets created in core engine - however response from core engine havent come back to Integration adaper due to some network/communication issues | No cant be replayed - Manual intervention will be required | INTG\_Non \_Replayable\_06 | Domain entity gets created in core engine - however response from core engine haven’t come back to Integration adaper due to some network/communication issues | Domain entity gets created in core engine - however response from core engine haven’t come back to Integration adaper due to some network/communication issues |
|  |  | Valid Business Error in the response payload from ICON/MF | Business Exception | ICON/MF returned valid business exception | No cant be replayed - Manual intervention will be required and event need to initiated again from source | INTG\_Non\_Replayable\_07 | Valid business exception from core engine | Business exception returned by [Ex: Icon/MF] and Error Code = [Business error code returned by ICON/MF ] and Error description=[Business Error description returned by iCON/MF] |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

Please refer below sample exception event

|  |
| --- |
| { "eventId": "22ef70fb-73e1-44e8-b023-069274c4f511", **-->Auto generated in the adapter would be used for lookup when replay triggered** "eventTimestamp": 1559545545140,  "timeZone": "UTC",  "payload": { "exceptionInfo": { "coreEventPayload": null**, --> Not applicable here** "name": "Adapter Exception",  "description": " Valid business exception from core engine ",  "exceptionClass": "ADAPTER",  "type": null,  "status": "NEW",  "errors": [ { "errorCode": "INTG\_Non\_Replayable\_07", **--> will be used for storing error code emitted by the adapter** "errorMessage": " Business exception returned by ICON and Error Code =ERR\_01 and Error description=partyid not found", **will be used for storing error message emitted by the adapter** "errorType": "Non Replayable Error" } ],  "exceptionContext": { "UUID": "4424424242", **🡪 Domain Entity Uber id** "transactionFlowType": "SUBS" },  "domainId": "4424424242", **🡪 Domain Entity Uber id** "exceptionID": "53ab4816-d7b7-4fd8-94c1-a88eb27e9757",  "links": [ ],  "PartyRole": { **--> Source Event Payload** "taOnshoreOffshoreIndicator": "onshore",  "UBERPtyId": "10",  "Type": "O",  "IsAmmendable": false,  "SingleSwingPricingParametersArray": [ { "SingleSwingPricingParametersOperationType": "",  "SingleSwingPricingParametersPricingThresholdMethodType": "" } --- --- },  "eventClass": "ADAPTER\_EXCEPTION\_EVENT",  "domainName": "PartyRole",  "eventHeader": { "previousEventId": null,  "sourceEventId": null,  "currentEventId": null,  "sourcePublisherId": null,  "sourceEventTimestamp": null,  "publisherId": null,  "mode": "MANUAL" },  "eventContext": { },  "eventName": " PartyRole\_PUBLISH",  "replayable": true,  "links": [ { "rel": "replay", --> **Replay Link** "href": "[http://localhost:8080/v1/replay/eventId=22ef70fb-73e1-44e8-b023-069274c4f511&eventName= PartyRole\_PUBLISH /accept](http://localhost:8080/v1/replay/eventId=22ef70fb-73e1-44e8-b023-069274c4f511&eventName=%20PartyRole_PUBLISH%20/accept)",  "hreflang": null,  "media": null,  "title": null,  "type": null,  "deprecation": null } ] } |

### 13.3.5 Configurations details (Application.yml/ConfigMap.yaml)

|  |  |  |
| --- | --- | --- |
| Sl # | Configuration Property | Sample values |
|  | spring.cloud.stream.bindings. invSchema-in .destination | nt\_mia\_party\_role\_icon\_out\_eventwriter\_event\_dev |
|  | spring.cloud.stream.bindings. invSchema-in .content-type | application/json |
|  | spring.cloud.stream.bindings. xrefSchema-out .destination | nt\_mia\_partyrole\_icon\_out\_crossref\_event\_dev |
|  | spring.cloud.stream.bindings. xrefSchema-out .content-type | application/json |
|  | spring.cloud.stream.bindings. error-out .destination | nt-mia-iu-mf-exception |
|  | spring.cloud.stream.bindings. error-out .content-type | application/json |
|  | spring.cloud.stream.kafka.binder.brokers | ut11241.ntrs.com:9092 |
|  | spring.cloud.stream.kafka.binder.auto-create-topics | "false" |
|  | Icon.create.partyrole.url | http://devirl10.ntrs.com:8081/ords/pdbirl01d/rest/fund/009 |
|  | spring.jackson.mapper.accept\_case\_insensitive\_properties | "true" |

### ICON Cross Ref ID Processor (Adapter-FuCom-Out-XREF)

### List of Major components

| Package name | Method name | Input | Output | URL pattern |
| --- | --- | --- | --- | --- |
|
| [com/ntrs/mia/integration/event/persist/topicreader/](https://git.ntrs.com:8443/projects/MIA/repos/external-xref/browse/src/main/java/com/ntrs/mia/integration/event/persist/topicreader/PersistListener.java) | CrossRefIdPersistListener.java | Json payload for Crud operation for Cross reference api | None | NA |
|  |  |  |  |  |

### Implementation Details

|  |  |  |
| --- | --- | --- |
|
|  | Business Logic | Input: JSON String – which will be input for Cross Ref Api create/update action  .  Output: None   1. Creat/Update/delete Internal and external ID mapping in the Cross reference api 2. Implement Hystrix while calling ICON api and cross ref api |
|  | Spring Boot Admin | Add dependencies for Spring Boot admin in POM  Config Map changes   * spring.boot.admin.client.url: [https://nt-spring-admin-server.ksd.ntrs.com](https://nt-spring-admin-server.ksd.ntrs.com/) * management.endpoints.web.exposure.include: "\*" * spring.boot.admin.client.instance.name adapter-fucom-out-xref spring.boot.admin.client.instance.service-url: adapter-fucom-out-xref [This should be K8 Service name] |
|  | Hystrix | Implement Hystrix while calling reference data api  In the Spring Boot Application add the below annotation (pseudo Code)  @EnableCircuitBreaker  In the method where we are invoking apis add the below annotation  @HystrixCommand(fallbackMethod = "iconCreatePRoleFallBack")  Add fallback method and apply fallback logic  public void iconCreatePRoleFallBack (XrefRequestPayload xref, Throwable e) {  log.info("iconCreatexrefFallBack #start: ");  } |
|  | Producer Side Message Compression | Not applicable as this will not publish any messages |
|  | Transaction handling | There is no requirement for business Transaction handling in this adapter. |
|  | Exactly Once Delivery | Would be done using consumer groups.. Configure Consumer Group, Minimum partition, instance-count, instance-index as shown in following application.yml file as red box  Pseudo configuration |
|  | TNT Crypt Encryption/Decryption | Channel encryption (Encrypt while putting message to Kafka and Decrypt while receiving message from Kafka)  Not applicable for Fund Utility as none of the fields for Fund described as PI Sensitive |
|  | Audit event | Not applicable for Integration adapter |
|  | K8 Service URL | [http://](http://adapter-fund-out-rdr/) adapter-fucom-out-xref |
|  | Common Library | Not applicable |
|  |  |  |

1. Source-To-Target Message Transformation

### Source to Target Mapping

This section details how incoming massage payload PartRole is mapped to ICON APIs request payload. Following table shows entities/sub entities in PartyRole and their attributes mapped to specific ICON APIs and their request payloads. It also details business logic for specific field transformation, static data look up logic, field format change logic or any other rule to transform source to target data.

<copy source to target mapping here …being worked upon in workshop>

| # | Target Entity | Attribute | Desc |  | Source entity | Attributes | desc | Rule |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | WriterProcessor(SOMTA Outbound) |  |  |  |  |  |  |  |

### ICON API contract

Please refer to ICON API contact [Ref-08]. This document contains ICON business entities, sub entities are their relationship. It provide list of attributes of an entity and corresponding back end table/column. API request/response payload contains same entities and attributes captured in this document.

### ICON API Definition (Swagger)

Please refer to the document ICON API contact [Ref-09]. This is standard swagger file for MF /ICON APIs, can be opened in appropriate tool to see detailed API signature, method, request/response and error code of the APIs and also to generate the target POJOs.

1. Traceability & Testing Strategy

This section covers how technical stores in the scope of this document can be traced back to key business use cases of Investor Utility. This is to validate weather desired FU use cases containing integration with ICON core system are actually considered in the integration scope.



### Traceability (Business Use cases to Tech Stories)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tech story | Precondition | Name | Icon API | IU Use case |
| MIA-166 | Investor PartyRole Utility sends Published PartyRole event | PartyRole Create in ICON | Party | •UC\_2\_New Client Setup - Investor Party Role creation  •IU23\_New Client Setup – Approval and if not eligible QCAml flow  •IU\_23 &IU\_24\_New Client Setup – Approval if QCAml  •IU\_24, 25, 26 \_New Client Setup - Approval if Manual Aml |
| 4 | Investor PartyRole Utility send Update event to PartyRole | PartyRole Update in ICON | Party |  |
|  |  |  |  |  |

### Testing Strategy

Separate comprehensive system integration test strategy document [Ref-10] is prepared to identify test cases for system integration testing to cover this specific IU to ICON integration scenario. Here are some key test scenarios to be tested from architect/designers point of view:

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Test case | Success criteria | Remark |
| 1 | Creating party role in core engine(Writer processor) | Internal ID available in the ICON response |  |
| 2 | Validate routing logic in reader router(Reader Router) | Correct technical attribute present in the Event and routing function returning success at the end |  |
| 3 | Update the external / internal ids to cross ref Api(Cross ref Id processor) | Response code 200 OK returned by Cross ref Api |  |

### Unit Testing

Unit test cases using JUNIT will be implemented for all major functions in the integration adapter’s components

Embedded KAFKA will be used for mocking KAFKA publisher/subscriber model

Mockito will be used for mocking the services/api calls like (Refer data api/cross ref id processing services, ICON API etc.)

Junit test cases are written for all individual adapter components starting from reader router to writer processor. Please refer below bitbucket links for all the junit test cases for all adapter component and please note junit test cases will keep changing while development will be in progress

|  |  |
| --- | --- |
| Adapter Component | BitBucket Url for junit test cases |
| Writer Processor | https://git.ntrs.com:8443/projects/MIA/repos/adapter-icon-partyrole-out-ewp/browse?at=refs%2Fheads%2Fdev |
| Transformer | https://git.ntrs.com:8443/projects/MIA/repos/adapter-icon-partyrole-out-mtf/browse?at=refs%2Fheads%2Fdev |
| ReaderRouter | https://git.ntrs.com:8443/projects/MIA/repos/adapter-partyrole-out-rdr/browse?at=refs%2Fheads%2Fdev |

1. Integration Adapter Repository List and Kafka Topics

### 16.1 Kafka Topics List

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S.No | Use case | Publisher | Subscriber | Topic name | Topic name (DEV) | Topic name (SIT) |
| 5 | Create Party Role | Investor Utility | PartyRole Outbound Integration ReaderRouter | nt\_mia\_party\_role\_ nt\_mia\_party\_role\_icon\_out\_transformer\_event | nt\_mia\_party\_role\_icon\_out\_transformer\_event\_dev | nt\_mia\_party\_role\_icon\_out\_transformer\_event\_sit |
|  |  | PartyRole Outbound Integration ReaderRouter | PartyRole ICON Outbound IntegrationTransformer | nt\_mia\_party\_role\_mf\_out\_transformer\_event | nt\_mia\_party\_role\_mf\_out\_transformer\_event\_dev | nt\_mia\_party\_role\_mf\_out\_transformer\_event\_sit |
|  |  | PartyRole ICON Outbound IntegrationTransformer | PartyRole ICON Outbound Integration Writer Processor | nt\_mia\_party\_role\_icon\_out\_eventwriter\_event | nt\_mia\_party\_role\_icon\_out\_eventwriter\_event\_dev | n nt\_mia\_party\_role\_icon\_out\_eventwriter\_event\_sit |
|  |  |  |  |  |  |  |
| 6 | Update Fund Party Role | Investor Utility | PartyRole Outbound Integration ReaderRouter | nt\_mia\_party\_role\_ nt\_mia\_party\_role\_icon\_out\_transformer\_event | nt\_mia\_party\_role\_ nt\_mia\_party\_role\_icon\_out\_transformer\_event\_dev | nt\_mia\_party\_role\_ nt\_mia\_party\_role\_icon\_out\_transformer\_event \_sit |
|  |  | PartyRole Outbound Integration ReaderRouter | PartyRole ICON Outbound IntegrationTransformer | nt\_mia\_party\_role\_mf\_out\_transformer\_event | nt\_mia\_party\_role\_mf\_out\_transformer\_event \_dev | nt\_mia\_party\_role\_mf\_out\_transformer\_event \_sit |
|  |  | PartyRole ICON Outbound IntegrationTransformer | PartyRole ICON Outbound Integration Writer Processor | nt\_mia\_party\_role\_icon\_out\_eventwriter\_event | nt\_mia\_party\_role\_icon\_out\_eventwriter\_event \_dev | nt\_mia\_party\_role\_icon\_out\_eventwriter\_event \_sit |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | Create/Update Party Role | Exception |  | nt-mia-iu-mf-exception | nt-mia-iu-mf-exception \_dev | nt-mia-iu-mf-exception \_sit |
|  |  |  |  |  |  |  |

### Bit Bucket Repository List

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Bitbucket Repo name | Description | Adapter | Source | Target | Project | Bit Bucket Urls |
| 11 | Adapter-PartyRole-Out-RDR | Outbound adapter subcomponent for reading Party Role event published by IU. (Message Reader) | PartyRole Icon Outbound Adapter/Party ICON Outbound Adapter | IU | ICON,MF | Investor Utility Core Integration | https://git.ntrs.com:8443/projects/MIA/repos/adapter-partyrole-out-rdr |
| 14 | Adapter-Icon-PartyRole-Out-MTF | Outbound adapter subcomponent for Message Transformationfrom NTISO to MultiFond for Party Role | PartyRole Icon Outbound Adapter | IU | ICON | ICON Integration | https://git.ntrs.com:8443/projects/MIA/repos/adapter-icon-partyrole-out-mtf |
| 15 | Adapter-Icon- PartyRole -Out-EWP | Outbound adapter subcomponent for Publishing to MultiFond for Fund Party Role | PartyRole Icon Outbound Adapter | IU | ICON | ICON Integration | https://git.ntrs.com:8443/projects/MIA/repos/adapter-icon-partyrole-out-ewp |
|  |  |  |  |  |  |  |  |

1. Microservices Design

### HTTP Response Codes

Any API exposed on SOM TA (e.g. Utilities APIs, ID Cross ref APIs) for internal or external consumption will return consistent HTTP response code in the response header. Following table list possible response codes for successful processing or error/failure cases. HTTP response status codes indicate whether a specific [HTTP](https://developer.mozilla.org/en-US/docs/Web/HTTP) request has been successfully completed or failed due to some internal/external problem. Following HTTP codes will be returned:

|  |  |  |  |
| --- | --- | --- | --- |
| Response | HTTP Method | Desription | Remark |
|
| 200 OK | GET | The resource has been fetched and is transmitted in the message body |  |
| PUT, POST | The resource describing the result of the action is transmitted in the message body. |  |
| 400 Bad Request | GET, POST, PUT | The server could not understand the request due to invalid syntax |  |
| 401 Unauthorized | GET, POST, PUT | Unauthenticated |  |
| 204 Not Found | GET | No content in the response |  |
|  |  |  |  |

1. Security Architecture

Please refer to section – 7 Integration adapter Security Architecture of common design [Ref-06]

1. Deployment

Please refer to section – 8 Environment, Infrastructure & Deployment Architecture [Ref-06]

1. Glossary & Abbreviations

|  |  |
| --- | --- |
| **Term** | **Definition** |
|  |  |
| **AML** | **Anti-money laundering** |
| **EDM** | **Enterprise Document Master** |
| **EU** | **European Union** |
| **FA** | **Fund Accounting** |
| **FX** | **Foreign Exchange** |
| **GDPR** | **General Data Protection Regulation** |
| **GIO** | **Global Invest One** |
| **NAV** | **Net Asset Value** |
| **MF** | **Multifonds** |
| **PACE** | **Performance and Analytics Engine** |
| **SoM** | **Strategic Operating Model** |
| **TA** | **Transfer Agency** |
| **TED** | **Transparency Engine Dashboard** |
| **TLM** | **Transaction Lifecycle Management** |
|  |  |

------------ End of the document ------------