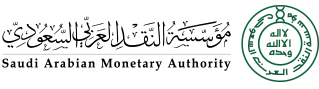


******

***Detailed Design Document***

Banking Enforcement Automation

Author Name: SBM Team

Version No:   Version 1.7 version

Issue Date:     18 Jul 2019

**Table of Contents**

[1 Introduction 7](#_Toc511312967)

[1.1 Project Overview 7](#_Toc511312968)

[1.2 Document Purpose 7](#_Toc511312969)

[1.3 Targeted Audience 7](#_Toc511312970)

[1.4 References 7](#_Toc511312971)

[2 Main Flow Scenario 8](#_Toc511312972)

[2.1 General Inquiry Services Sequence Diagrams 8](#_Toc511312973)

[2.1.1 Scenario 1: 8](#_Toc511312974)

[2.1.2 Scenario 2: 11](#_Toc511312975)

[2.1.3 Header Message Correlation flow 14](#_Toc511312976)

[3 IIB Services Design 25](#_Toc511312997)

[3.1 Integration Bus Service High Level Diagram 25](#_Toc511312998)

[3.2 Naming Convention 26](#_Toc511312999)

[3.3 Header Fields 27](#_Toc511313000)

[3.4 Audit logging 28](#_Toc511313001)

[3.5 Exception Handling 29](#_Toc511313002)

[3.6 RP Inbound Request (Exposed to Data Power) 30](#_Toc511313003)

[3.6.1 Service Interface 30](#_Toc511313004)

[3.6.2 Message Flow Design 30](#_Toc511313005)

[3.6.3 Request & Response Mapping 31](#_Toc511313006)

[3.7 FI Inbound Callback (exposed to Data Power) 31](#_Toc511313007)

[3.7.1 Service Interface 31](#_Toc511313008)

[3.7.2 Message Flow Diagram 31](#_Toc511313009)

[3.7.3 Request & Response Mapping 32](#_Toc511313010)

[3.8 BPM Outbound Gateway (Exposed Internally) 33](#_Toc511313011)

[3.8.1 Service Interface 33](#_Toc511313012)

[3.8.2 Message Flow Diagram 33](#_Toc511313013)

[3.8.3 Request & Response Mapping 34](#_Toc511313014)

[3.9 Supplementary sub flows 34](#_Toc511313015)

[3.9.1 Load & Cache Data Sub flow Diagram 34](#_Toc511313016)

[3.9.2 Exception Handling Sub flow Diagram 35](#_Toc511313017)

[3.9.3 MQ Invocation Logic Sub flow Diagram 35](#_Toc511313018)

[3.9.4 Webservice Invocation Logic Sub flow Diagram 36](#_Toc511313019)

[3.10 Call ODM Rule Service 37](#_Toc511313021)

[3.10.1 Service Interface 37](#_Toc511313022)

[3.11 Email Notification 38](#_Toc511313023)

[3.12 Retry Mechanism 39](#_Toc511313024)

[3.13 Mock Flow for Soap 40](#_Toc511313025)

**Record of changes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Change details** |
| 28 January 2018 | 0.9 | SBM team | Draft version |
| 24 February 2018 | 1.0 | SBM team | Initial version |
| 28 February 2018 | 1.1 | Mohammed Gawad | Update BPEL changes |
| 01 March 2018 | 1.2 | AbdelMoneim Azzaz | Re-organize the structure of the document sections |
| 26 March 2018 | 1.3 | Haitham Eltaweel | IIB Services Design changes |
| 27 March 2018 | 1.4 | Haitham Eltaweel | IIB Retry mechanism added |
| 29 March 2018 | 1.5 | Haitham Eltaweel | Queue names updated in Integration bus service high level diagram |
| 12 April 2018 | 1.6 | Haitham Eltaweel | IIB Retry mechanism updated |

**Distribution List**

|  |  |
| --- | --- |
| **MOJ** |  |
| **SAMA UAT/Testing** |  |
| **Solution Delivery** |  |
| **Project Sponsor** |  |
| **Project Manager** |  |
| **Architecture Solution Unit** |  |

**Approvals List**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Signature** | **Date** |
|  | SAMA Architecture Representative |  | Date of Approval |
|  | SAMA Solution Delivery |  |  |
|  | SAMA SOC team |  |  |
|  |  |  |  |
|  |  |  |  |

*Approvals can be obtained by e-mails, or document sign-off*

# Introduction

## Project Overview

The main goal of the project is to handle a set of required Inquiry services Requests coming from external governmental entities like (like MJO "ministry of justice", MOI Ministry of interior, and, etc …) to a common SAMA infrastructure and processes for handling and responding to these incoming requests which communicate with the other bank all over Saudi Arabia to handle these requests, also to shorten the overall process duration by automating as many activities as possible avoiding unnecessary human reviews and informing department responsible for bank execution monitoring in case there is no timely response from a bank.

## Document Purpose

This document has the technical detailed Design for SAMA inquiry services [WATHEEQ]and processes in the Areas of IIB, WSRR and DataPower

## Targeted Audience

The requirements are targeted to the following stack holders:

* SAMA technical team to validate the design against the business requirements.
* SBM architects to validate the design against the overall architecture.
* SBM development team who will implement processes.

## References

|  |  |  |  |
| --- | --- | --- | --- |
| Document | Version | Date | Owner |
| Process Flow Design Document |  |  |  |
| Use Cases Document |  |  |  |
| Business Requirement Document |  |  |  |
| Architecture Document |  |  |  |
| Logical Data Model Document (ERD) |  |  |  |

# Main Flow Scenario

## General Inquiry Services Sequence Diagrams

### Scenario 1:

General Inquiry happy path scenario through SAMA B2B (web-service enabled FI) without human intervention, this scenario represents 90% of the General Inquiries requests

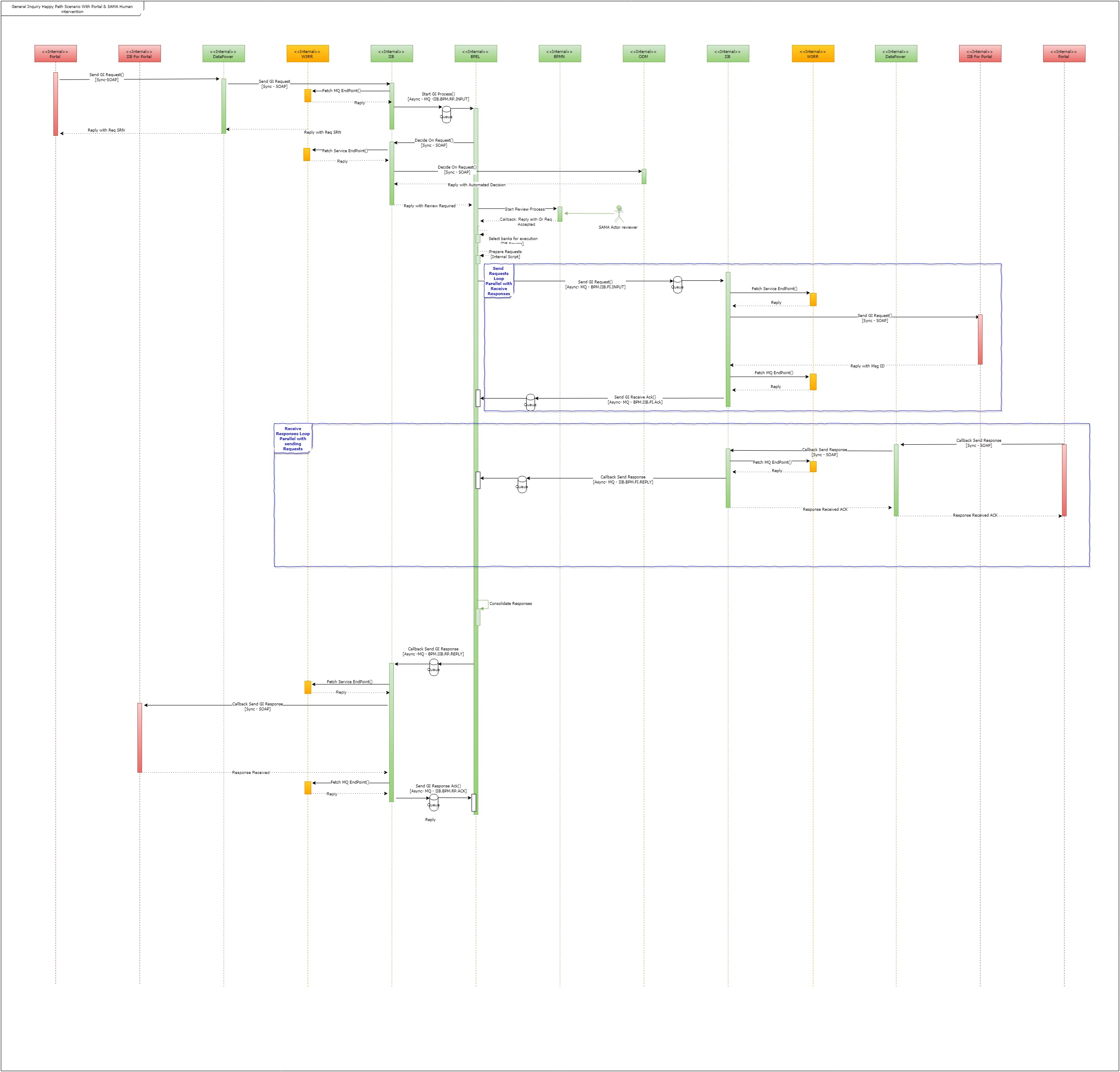
**

***Flow Sequence Description:***

1. *The Flow scenario starts from the* ***MOJ*** *external entity which request a one atomic request for one of the GI requests (getAccountInfo, getAccountBalanceInfo, getDepositInfo, getSafeInfo, getLiabilitiesInfo) through its internal application which calls our webservices defined in the* ***DataPower*** *Layer*
2. *When the* ***DataPower*** *receive the requests as a proxy layer it makes a client certificate check and validate the received XML request against XSD schema which will be passed in our scenario*
3. *The request will be sent to the* ***IIB*** *which plays the ESB layer for all SAMA services, the request will be audited and logged, then it will start the* ***BPEL*** *GI**process by sending a message over the queue which contains the SRN generated number*
4. *The* ***BPEL*** *Business Process will continue its work asynchronously after sending the response back directly, the first step after saving the request record to the Database is checking if the request needs review from SAMA employee this checking will be done by calling the ODM Rule, or it will go direct to financial institutes if no review is needed (In this scenario no review is needed)*
5. *Then next step will be selecting financial institutes required for handling this request by internal service logic, then preparing these requests to be sent to each financial institute.*
6. *Through a loop in the BPEL Process the requests will be sent to the* ***IIB*** *to forward to the DataPower then to the Required* ***B2B financial institutes*** *to handle the request.*
7. *Once any of the* ***B2B financial institutes*** *receive the request it will reply with a request ID and reply to the* ***DataPower*** *with it as an acknowledgment, then it will be sent back to the* ***IIB*** *which will reply to the caller* ***BPEL*** *Process over queue.*
8. *The* ***BPEL*** *Process stores the request sent time in the Database and start the SLA calculation for each sent request.*
9. *Once the calculation of the SLA started for any of the sent request the BPEL process should make a call-back method ready for receiving the response and that sent request*
10. *Once the* ***B2B bank*** *finishes the**GI request it will respond back to the* ***DataPower*** *through its call-back method****.***
11. *The* ***Datapower*** *will respond back to the* ***IIB*** *call-back method which will pass the response to the* ***BPEL*** *process over the queue.*
12. *The* ***BPEL*** *replies back with response received acknowledgment which will be propagated* ***B2B Bank*** *through the* ***IIB*** *and* ***Datapower*** *internal layers.*
13. *The* ***BPEL*** *will stop the SLA for the received response and will record the received time in the Database.*
14. *After receiving all the responses from the selected* ***Banks,*** *the* ***BPEL*** *will send the response to the* ***IIB*** *which will forward to the* ***DataPower*** *to send to the Original* ***MOJ*** *requester**through a call back method.*
15. *The* ***MOJ*** *replies with response received acknowledgment to the* ***Datapower*** *which will propagate to the* ***IIB*** *then to the* ***BPEL*** *Process which record the response sent time in the DB then end the process.*

### Scenario 2:

General Inquiry request happy path scenario through SAMA internal portal (Non- web-service enabled FI) and with SAMA reviewer’s human intervention.



***Flow Sequence Description:***

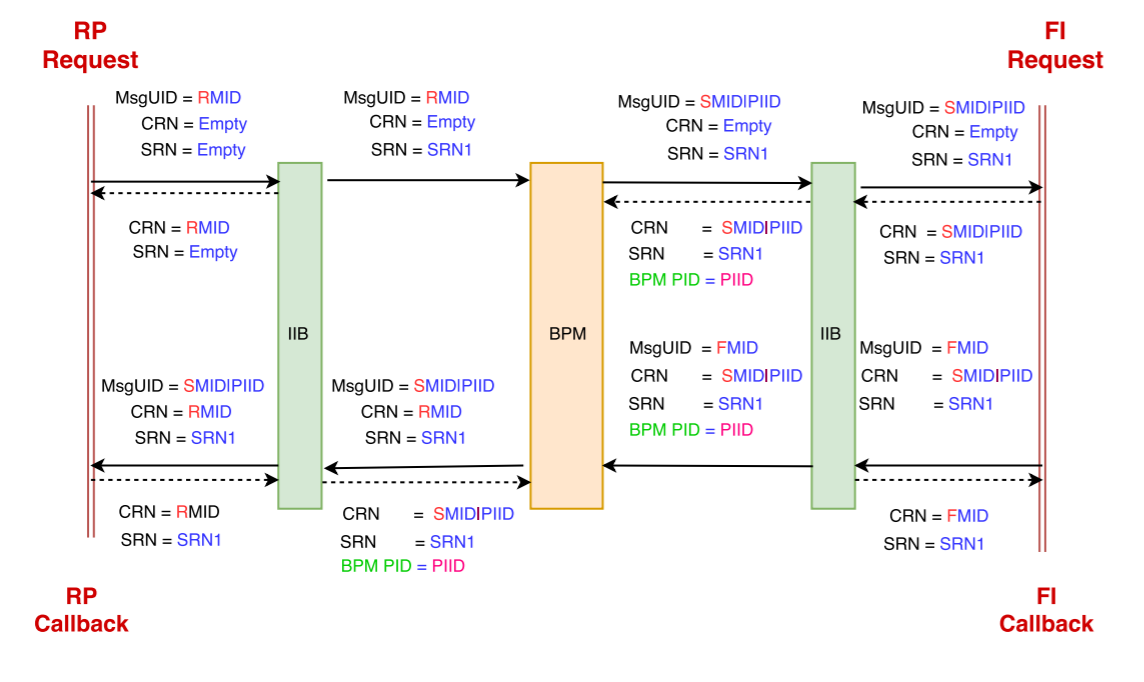
1. *The Flow scenario starts this time by the* ***MOI*** *employee through internal* ***SAMA portal*** *in the DMZ layer which exposed to* ***MOI*** *external entity to make a one atomic request for one of the GI request.*
2. *The request came from* ***SAMA portal*** *to the* ***DataPower*** *to send the required GI Request to be handled by SAMA and the desired banks.*
3. *When the* ***DataPower*** *receive the requests as a proxy layer it makes a client certificate check and validate the received XML request against XSD schema which will be passed in our scenario*
4. *The request will be sent to the* ***IIB*** *which plays the ESB layer for all SAMA services, the request will be audited and logged, the it will start the* ***BPEL*** *GI**process by sending the request with the generated SRN to it.*
5. *The* ***BPEL*** *Business Process will continue its work asynchronously after sending the response back directly, the first step after saving the request record to the Database is checking if the request needs review from SAMA employee or this by calling the ODM Rule, which will be review is required in our case.*
6. *The* ***BPEL*** *process will start another* ***BPMN*** *process in the for dropping human task for SAMA reviewer through BPM Portal.*
7. *After the reviewer send his complete his task in the BPM portal the* ***BPMN*** *Process will call back the BPEL process with the review result which will be in our scenario request is accepted*
8. *Then next step will be selecting banks required for handling this request by internal service logic, then preparing these requests to be sent to each bank.*
9. *Through a loop in the BPEL Process the requests will be sent to the* ***IIB*** *to forward to the IIB-Portal service which will save the request which will be consumed through SAMA portal by the banks which has not B2B systems.*
10. *The* ***BPEL*** *Process stores the request sent time in the Database and start the SLA calculation for each sent request after receiving the acknowledgment from the* ***IIB****.*
11. *Once the calculation of the SLA started for any of the sent request the BPEL process should make a call-back method ready for receiving the response and that sent request.*
12. *For the Portal Banks, it will pull the requests from the Database by using service in the* ***IIB*** *which will be invoked through the* ***DataPower***
13. *Once the* ***Portal bank*** *finishes the**GI request it will respond back to the* ***DataPower*** *through its call-back method****.***
14. *The* ***Datapower*** *will respond back to the* ***IIB*** *call-back method which will pass the response to the* ***BPEL*** *process call-back method over queue.*
15. *The* ***BPEL*** *will stop the SLA for the received response and will record the received time in the Database.*
16. *After receiving all the responses from the selected* ***Banks,*** *the* ***BPEL*** *will send the response to the* ***IIB*** *which will forward to the* ***DataPower*** *to send to the Original* ***MOI*** *requester**through* ***SAMA Portal****.*
17. *The* ***SAMA Portal*** *replies with response received acknowledgment to the* ***Datapower*** *which will propagate to the* ***IIB*** *then to the* ***BPEL*** *Process which record the response sent time in the Database then end the process.*

***Note the following general rules for request flow****:*

1. All the URL services which will be invoked from the IIB layer will be fetched from the WSRR.
2. The DataPower will act as a proxy in the DMZ network layer.
3. The communication between the IIB and the BPEL process will be through queue avoid any concurrency problem, message delivery assurance and to avoid any timeout in the caller side.
4. The DataPower will fetch the external systems URL from the WSRR for the PR and FI
5. If request comes to the Data power for a specific operation from an external entity, the DataPower will check from the WSRR to check if it has the privileges to access this operation or not before sending it to the IIB.

### Header Message Correlation flow

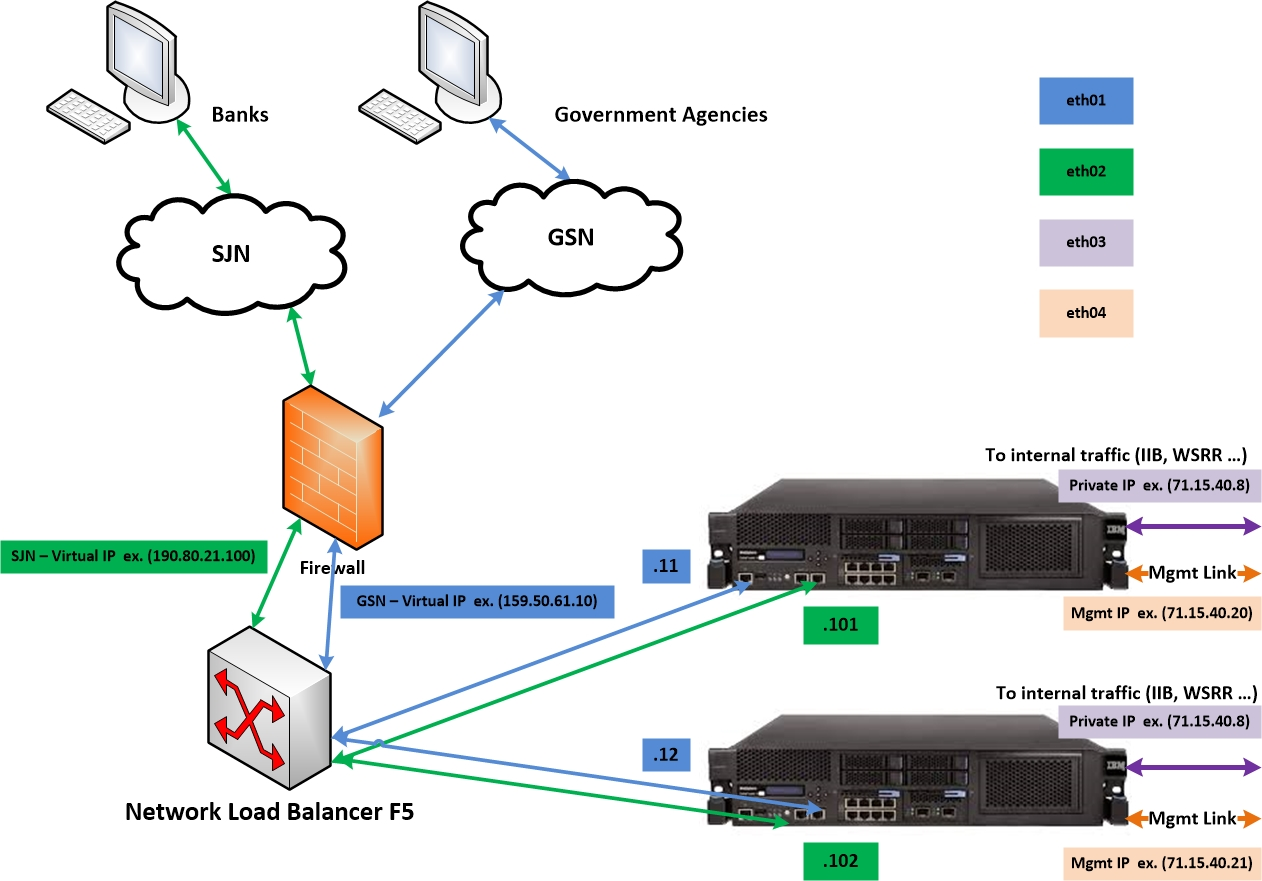
The following sequence diagram describes part of the message header values which related to the request correlation



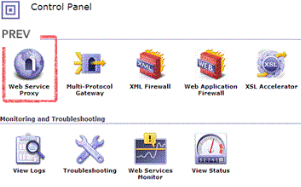
|  |  |
| --- | --- |
| Field | Discerption |
| MsgID | This will be a universal message identifier from the source, which will be used for duplicate elimination |
| CRN | Which contains the MsgUID of the original request message, It is used for the Callback request messages |
| SRN | SAMA creates this code. Tracking reference between Partners of each RP "Service Request”. In of case of FI, it is used to link many requests together; for example, if one service requested by RP and needs more than one service with FI, this code will correlate them |
| BPMPID | Attribute between the BPM & IIB which used to correlate different sent messages to one BPM Process Instance |

# DataPower Services Design

This figure shows the SAMA Datapower architecture overview



## Web Service Proxy



To external parties IBM DataPower will be front-facing device. It will accept incoming requests and forward them to the back-end services.

The main mechanism how this is achieved by using the Web Service Proxy objects. Web Service Proxy objects will be defined by using the WSDLs provided by the IBM Integration Bus services.

Web Service Proxy will perform the following:

* 1. Accept incoming HTTPS (TLS) traffic.
  2. Read client certificate from the TLS connection
  3. Identify the client (extract DN)
  4. Forward DN to LDAP and get validate.
  5. Validate the request by the XML schema
  6. Forward the request with client information to the back-end service
  7. Wait for the reply and forward reply to the client

Web Service proxy also has other security features such as XML firewall, preventing malicious attacks. also, DoS attacks can be blocked by setting max request rate from host and other properties.

* Data Power will accept incoming requests and forward them to the IIB.
* Web Service Proxy objects will be defined by using the WSDLs provided by the IIB services.
* External party will authenticate itself by the means of providing certificate in TLS two-way handshake.
* Data Power is responsible to terminate client SSL connection. Only clients with trusted certificates (SAMA CA generated certificates) will be allowed to connect to Data Power.
* Data Power will authenticate clients using following sequence:
  + Extracts identity from client certificate (e.g. using dp:auth-info('ssl-client-subject', 'ldap-strict') function)
  + If certificate DN is not in the expected list Data Power rejects the request with appropriate error code (HTTP error code 401).

## Security

### Transport Security

All communication incoming to the SAMA system should be secured on transport layer by using TLS 1.2 secure protocol. TLS should be used to secure HTTP traffic, incoming and outgoing. Also, it should be required to block unwanted HTTP connections by using two-way authentication.

### Two-way authentication

Service calls from external zone should be secured by two-way authentication.

This should work in following context:

1. SAMA acts as CA authority and creates certificates
2. Certificates are created for the external parties.
3. External party stores certificate in keystore
4. All external parties have SAMA CA certificate in truststore
5. When new connection is about to be established external party is required to send certificate
6. External party certificate is verified and only if it is trusted connection could be established



Two-way authentication

### AAA

Only authenticated systems could call SAMA services. External party will authenticate itself by the means of providing certificate in TLS two-way handshake.

IBM DataPower (IDP), which serves as edge security device, is responsible to terminate client SSL connection. Only clients with trusted certificates (SAMA CA generated certificates) will be allowed to connect to IDP.

IDP will authenticate clients using following sequence:

* IDP extracts identity from client certificate(e.g. using dp: auth-info('ssl-client-subject', 'ldap-strict') function) and partner id.
* If certificate DN and Partner id not in the expected list IDP rejects the request with appropriate error code (HTTP error code 401).

After authentication next step is authorization.

DataPower will check in WSRR if the caller external party has authorization to access the target requested service

The request will be audited in **WATHEEQ** database

## Routing Framework

Datapower open 443 port for external parties. We need to implement routing gateway to route incoming requests to our webservice proxy.

## Logging

### On device Logging

The IBM DataPower Gateway provides a system log, and audit log, and various log targets. By default, the IBM DataPower Gateway provides the following logs.

* System log
  + Contains system-wide messages for the default domain, and contains the domain-specific messages for an application domain. The system log contains messages at or above the configured priority level. When the system log reaches 5120 KB, its content is moved to a rotation. The IBM DataPower Gateway maintains four rotations.
* Audit log
  + Contains messages about changes to the configuration of the IBM DataPower Gateway and files that it stores. The audit log contains all audit messages. Its configuration cannot be changed.

Beyond the system log, you can configure log targets to capture log messages that are based are various criteria.

### Off device Logging

The DataPower device has a finite amount of space to hold larger than average log files or long-term logging needs in production environment. DataPower only keeps a limited number of log files (the default is 3 files) in the file system in a rotational basis.

To resolve this problem off-device can be configured.

Preferable method is to configure stand-alone syslog server that will log the received events from DP device(s).

There should be a server daemon to listen and capture the DataPower devices events.

On DataPower we will create log target to send events to a syslog destination.

When needed, logs can be investigated on the syslog server without needing an access to DataPower. Also, there should be scheduled job to perform log housekeeping on syslog server

## Error Framework

There are many circumstances where an error can occur while processing security information.

* Invalid or unsupported type of security token, signing, or encryption
* Invalid or unauthenticated or un-authenticatable security token
* Invalid signature
* Decryption failure

These can be grouped into two classes of errors: unsupported and failure. For the case of unsupported errors, the receiver MAY provide a response that informs the sender of supported formats, etc. For failure errors, the receiver MAY choose not to respond, as this may be a form of Denial of Service (DOS) or cryptographic attack. We combine signature and encryption failures to mitigate certain types of attacks.

If a failure is returned to a sender then the failure MUST be reported using SOAP's Fault mechanism.

## Log Target

The logging subsystem on DataPower SOA Appliances is based on a publish-subscribe concept that enables distribution of selected messages to various protocols and destinations; message selection is a user-configurable process that can select broad message categories and priorities but can also be very granular if necessary.

Event messages can be generated by anything on the appliance, including a hardware-level device monitor, a service processing an application transaction, or a stylesheet logging a custom error message. Obviously, some of these items are more critical than others, and some may only be of interest to a select group of users; some messages may not be of any interest at all.

## Email

SMTP Server will be configured in DataPower for sending failure notification.

## Secure Backup

Secure backup will be executed on timely fashion.

## Deployment Framework

Common deployment policy will be implemented

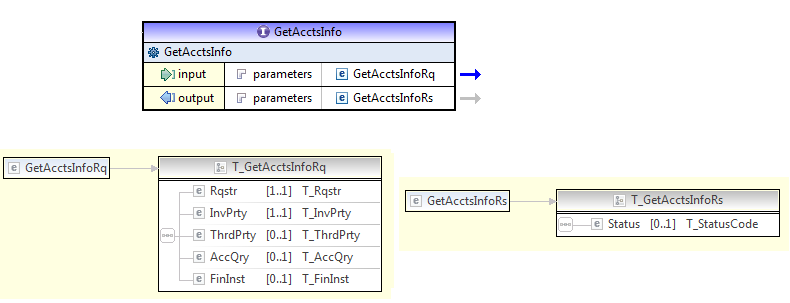
## DNS Configuration

A Domain Name System (DNS) server is required to be configured for the Appliance as DNS lookup services are used for communication.

## Get Accounts Info Service (Exposed to RP)

* **Service Interface**

This service exposed externally to RP to get info about accounts from FIs.

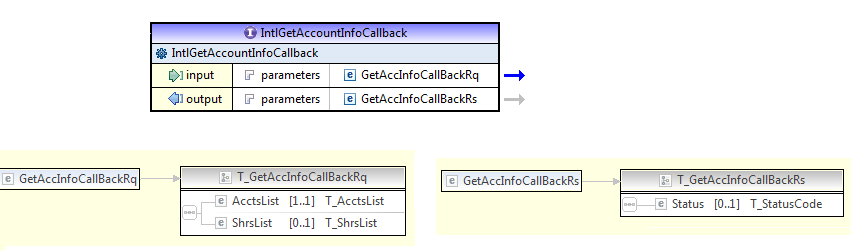


* **Service Flow description**
* Data Power Receive the incoming HTTPS (TLS) RP request.
* Read client certificate from the TLS connection
* Identify the client (extract DN)
* If valid certificate Data Power
* Data Power make schema validation
* Data Power calls WSRR and makes query to find out the service endpoint (URL for the IIB service) and check if the caller(RP) has access to consume this service or not
* Data Power calls IIB to forward the RP request
* Data Power forward the IIB response to the RP
* **Service Exceptional Flow**

## Inl Get Accounts Info Call Back Service (Exposed Internally)

* **Service Interface**

This service exposed internally which IIB will call after receiving FI consolidated response from BPEL to send to the RP

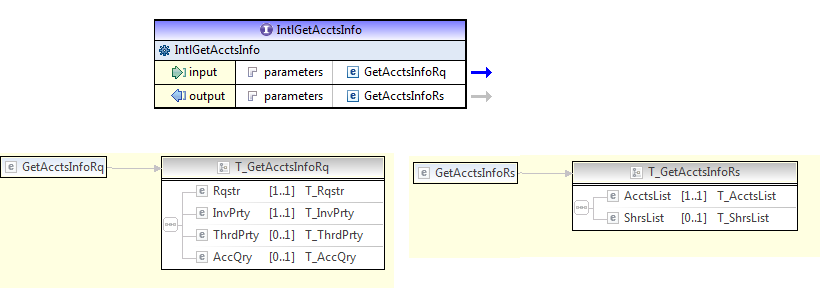


* **Service Flow description**
  + Data Power receives the request from IIB
  + Data Power add transport security layer
  + Data Power calls WSRR and makes query to find out the service endpoint (URL for the RP service)
  + Data power call a RP service to send the FI response which received from IIB
  + Data Power forward the RP Service response to IIB
* **Service Exceptional Flow**

## Intl Get Accounts Info Service (Exposed Internally)

* **Service Interface**

This service exposed internally which IIB will call to send the RP request to FI.



* **Service Flow description**
  + Data Power receives the request from IIB
  + Data Power add transport security layer
  + Data Power calls WSRR and makes query to find out the FI service endpoint (URL for the FI service)
  + Data power call the FI to send the RP Request which received from IIB
  + Data Power forward the FI response to IIB
* **Service Exceptional Flow**

## Get Accounts Info Call Back Service(Exposed to FI)

* **Service Interface**

This service exposed externally to receive the response from FIs for the RP request

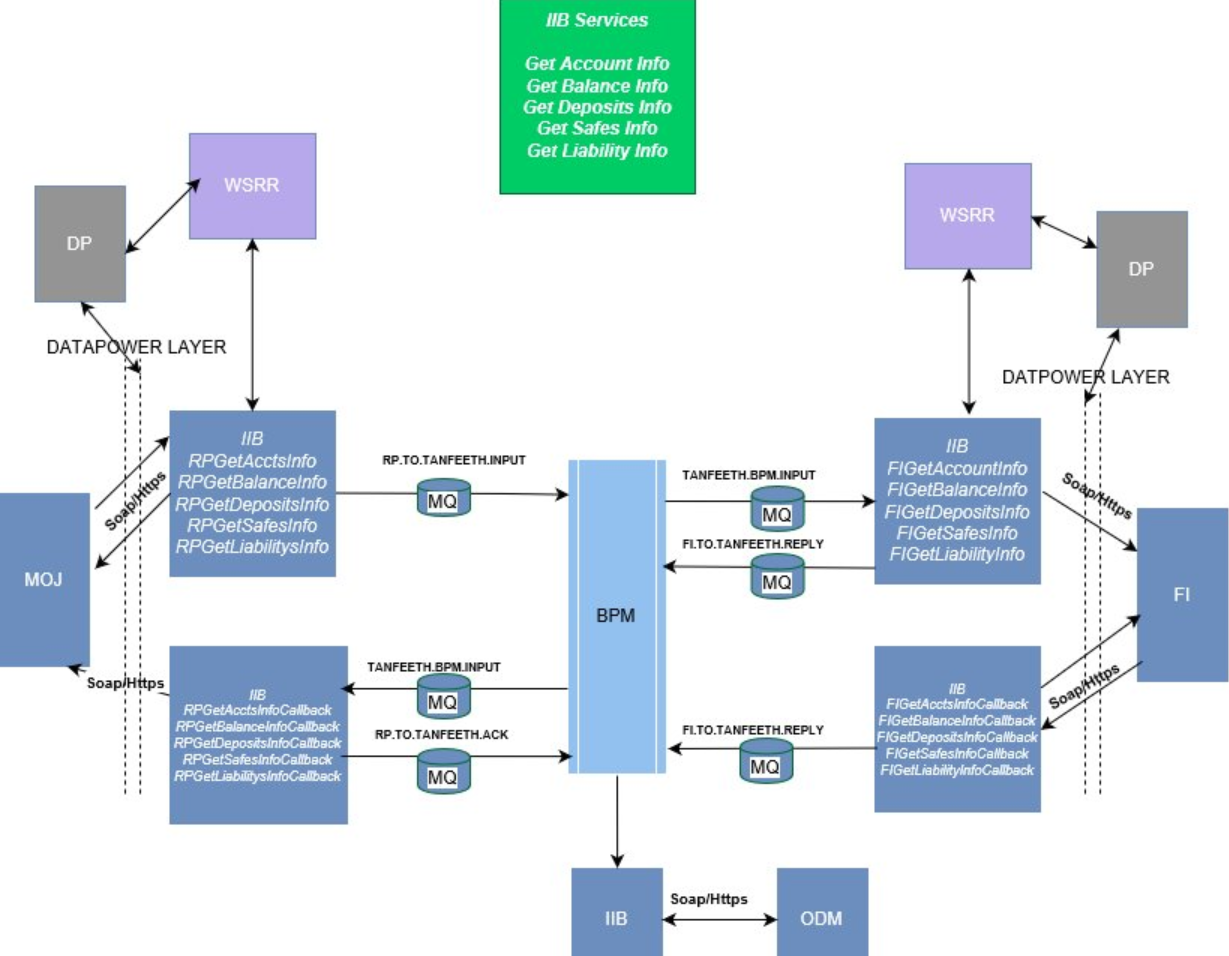


* **Service Flow description**
  + Data Power Receive the incoming HTTPS (TLS) FI request.
  + Read client certificate from the TLS connection
  + Identify the client (extract DN)
  + Data Power make schema validation
  + Data Power calls WSRR and makes query to find out the service endpoint (URL for the IIB service) and check if the caller(FI) has access to consume this service or not.
  + Data Power call IIB to forward the FI request
  + Data Power reply the FI by the IIB Service response
* **Service Exceptional Flow**

**Note:** The Other services Get Balance Info, Get deposit Info, Get Safe Info , and Get Liabilities Info it will follow the same steps as in Get Account Info service

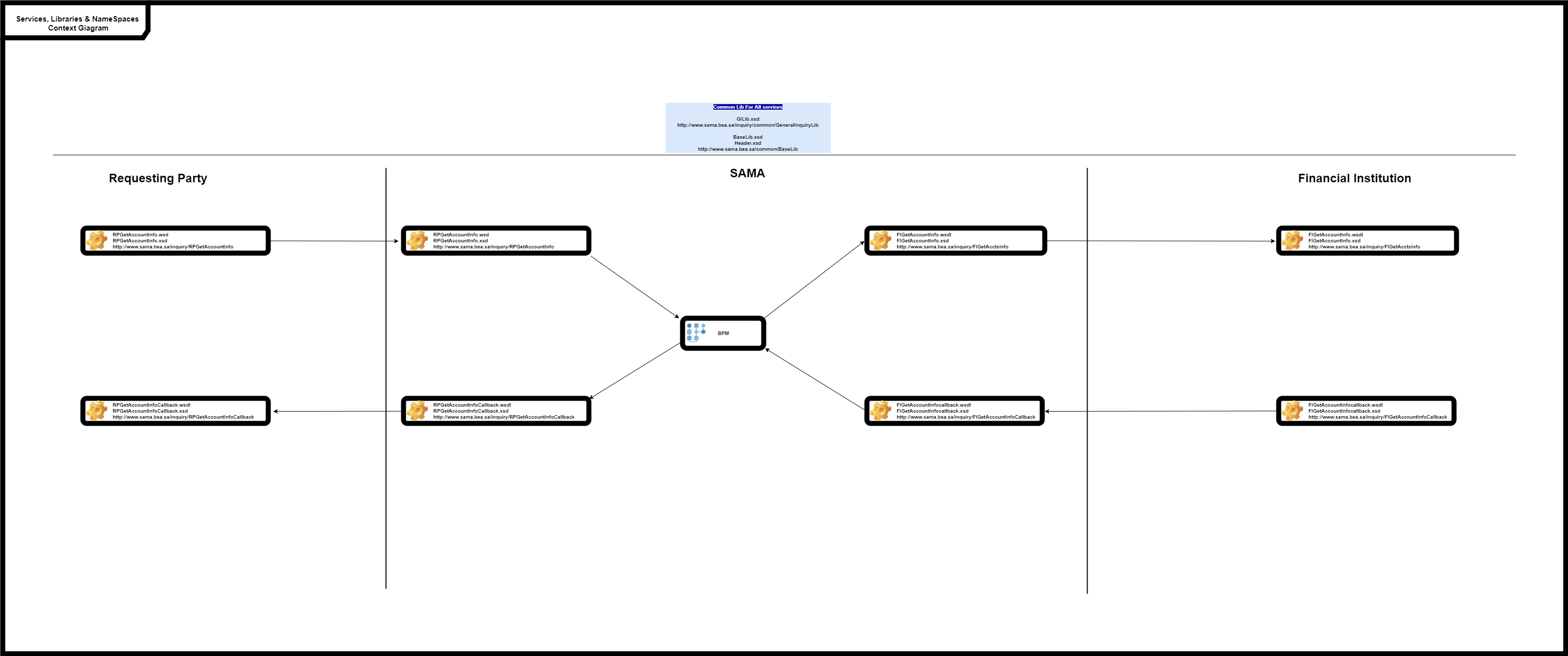
# IIB Services Design

## Integration Bus Service High Level Diagram

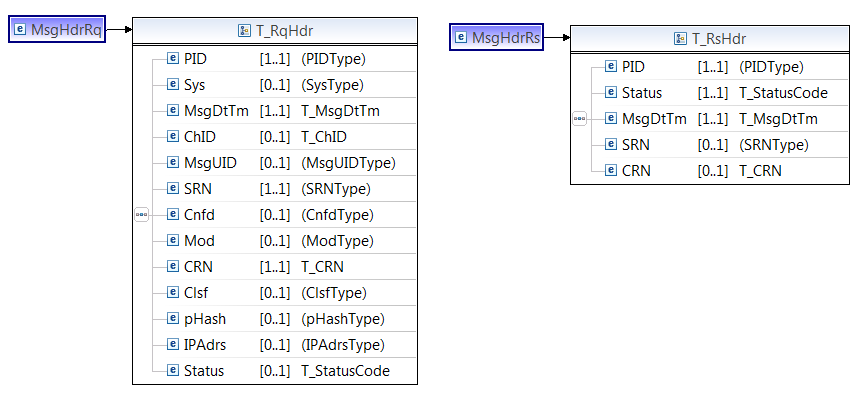


## Naming Convention

|  |  |  |  |
| --- | --- | --- | --- |
| **Name Spaces** | | | |
| 1 | Requesting Party Side | http://www.sama.bea.sa/inquiry/RPServicexxInfo | |
| http://www.sama.bea.sa/inquiry/RPServicexxCallback | |
| 2 | Financial Institution Side | http://www.sama.bea.sa/inquiry/FIServicexx | |
| http://www.sama.bea.sa/inquiry/FIServicexxcallback | |
| 3 | Common for servicexx | http://www.sama.bea.sa/inquiry/Servicexx | |
| 4 | Common for GI | http://www.sama.bea.sa/inquiry/common/GeneralInquiryLib | |
| 5 | Common | http://www.sama.bea.sa/common/BaseLib | |
| 6 | Common Header | http://www.sama.bea.sa/common/Header | |
| **Services Names** | | | |
| 1 | Requesting Party Side | Adding  **RP** perfix | RPServicexx |
| RPServicexxCallback |
| 2 | Financial Institution Side | Adding **FI** perfix | FIServicexx |
| FIServicexxcallback |
| **XSD File Names** | | | |
| 1 | Requesting Party Side | Adding  **RP** perfix | RPServicexx |
| RPServicexxCallback |
| 2 | Financial Institution Side | Adding **FI** perfix | FIServicexx |
| FIServicexxcallback |
| 3 | Common Part | N/A | Servicexx  BaseLib  GILib  Header |



## Header Fields

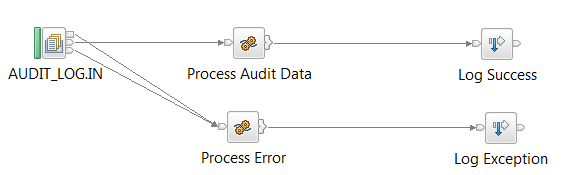


## Audit logging

This is a sub flow designed to collects audit messages and saves theirs content to the appropriate DB tables. Later audit messages can be searched and inspected via DB tool.

The message ID column in DB must be same for entire flow transaction for each service.

***Application Name:*** *AuditLoggerApp*



* + - **Service Flow description**
* IIB receive message in AUDIT\_LOG.IN queue
* To auditing (logging) to work the following must be in place.
  + AUDIT\_LOG, AUDIT\_LOG\_DATA Database tables created
  + ODBC data source accessible by IIB
  + Business application that creates monitoring events
  + Logging queue and durable subscription created on the MQ
  + Business application and AuditLoggerApp deployed
  + Monitoring turned on for business application turned on
* After deploying business application, one must activate the monitoring, by default it is disabled. This can be done in a few ways, one is from the console using command (other is using web console).
* It logs all Info level trace logs into ExecutionTraceLog.log file
* Implemented multiple levels, namely DEBUG, INFO, WARN, ERROR by using Create Audit Message entries.

## Exception Handling

**Extract Exception**: Extract of ESQL that has been coded for a Compute node to loop through the exception list to the last exception description and extract the error number, Text. This error relates to the original cause of the problem and normally provides the most precise information.

**Understanding causes of exceptions:** Connect the Failure terminal of any node to a sequence of nodes that processes the node's internal exception (the Failure flow).Connect the Catch terminal of the input node or a TryCatch node to a sequence of nodes that processes exceptions that are generated beyond it (the Catch flow). Insert one or more TryCatch nodes at specific points in the message flow to catch and process exceptions that are generated by the flow that is connected to the Try terminal.

Ensure that all messages that are received by an MQInput node are processed within a transaction or are not processed within a transaction.

* When an exception is detected within a node, the message and the exception information are propagated to the node's Failure terminal (Diagnostic information is available in the Exception List).
* If the node does not have a Failure terminal or if it is not connected, the broker throws an exception and returns control to the closest previous node that can process the exception. This node can be a Try Catch node (Root & Local Environment are reset to the values they had before) or the MQ Input node.
* If the catch terminal of the MQ Input node is connected, the message is propagated there (Exception List entries are available; Root & Local Environment are reset to the values they had before). Otherwise it is not connected, the transactionality of the message is considered.
* If the message is not transactional, the message is discarded. Otherwise, if it is transactional, the message is returned to the input queue, and it is read again, whereupon the backout count is checked.
* If the backout count has not exceeded its threshold, the message is propagated to the output terminal of the MQInput node for reprocessing. Otherwise if it is exceeded & if the failure terminal of the MQInput node is connected then the message is propagated to that path. (Root is available but ExceptionList is empty)
* If the failure terminal of the MQInput node is not connected, the message is put on an available queue, in order of preference; message is put in backout queue, if one is defined; otherwise in dead-letter queue, if one is defined. If the message cannot be put on either of these queues, it remains on the input queue in a retry loop until the target queue clears. (It also records the error situation by writing errors to the local error log)

## RP Inbound Request (Exposed to Data Power)

### Service Interface

This service exposed to Data Power to receive and process any of the five RP inquiry request services: GetAccountsInfo, GetBalanceInfo, GetDepositInfo, GetLiabilityInfo, GetSafesInfo

### Message Flow Design

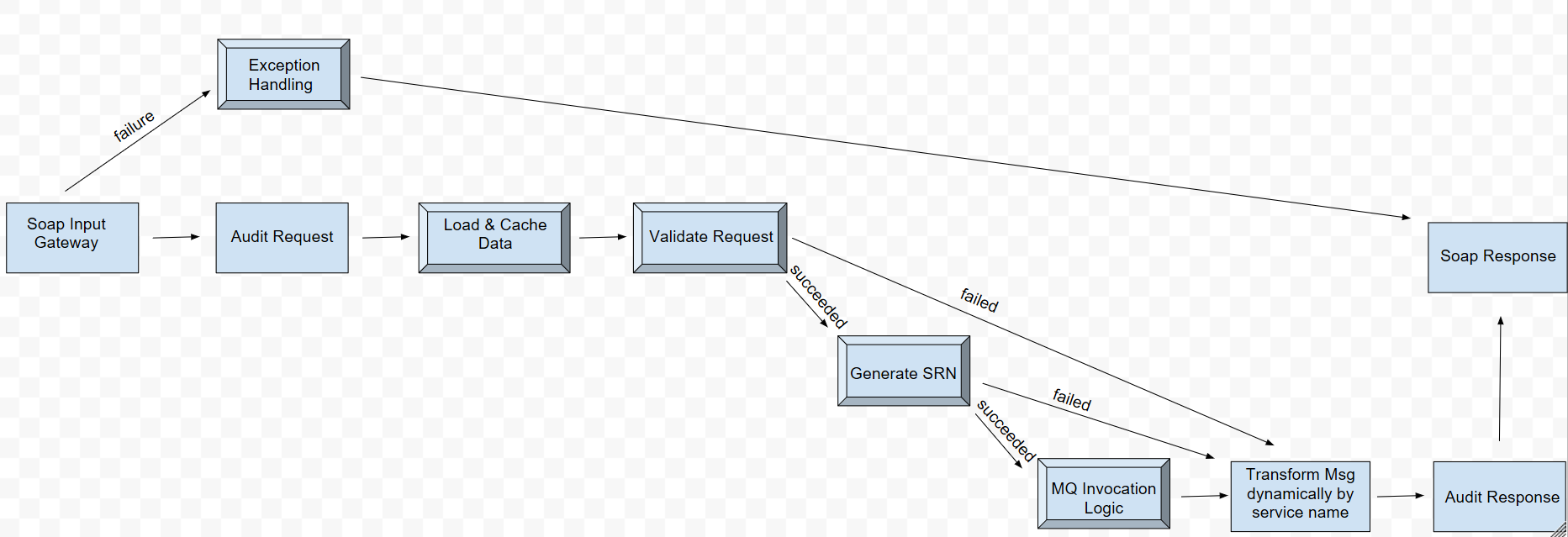


Figure 1 RP Inbound Request Flow

* + **Message Flow description**

1. Soap Input Node receives the RP request message from Data power.
2. Audit incoming message.
3. Load and cache data sub flow. Refer to section 4.9.1
4. Validate request sub flow
   1. Check partner Id existence in LOVs. Check message Id duplication. Perform dependency validations.
   2. If validation Succeeded, Generate SRN subflow will be called.
   3. If validation failed, the message will be transformed dynamically to failed soap response. Specific esql mapping file will be called at run time using the service name (xpath to the soap message body).
5. Generate SRN sub flow
   1. If SRN generation failed, the message will be transformed dynamically to failed soap response. Specific esql mapping file will be called at run time using the service request name.
   2. If SRN generation succeeded, MQ Invocation logic sub flow will be called.
6. MQ Invocation Logic sub flow. Refer to section 4.9.3
7. Generate RP soap response dynamically. Specific esql mapping file will be called at run time using the service name. Response will have status code, CRN (from Message UID of the RP request) and SRN.
8. Audit outgoing response.
9. Send the soap response
10. In case of any technical error (i.e. Webservice timeout, Database or WSRR connection error, etc.), exception will be captured, propagated to first node then sent to Exception Handling sub flow
11. Exception Handling sub flow. Refer to section 4.9.2

### Request & Response Mapping

Request and response mapping will be performed dynamically by calling the corresponding esql mapping file based on the service name.

## FI Inbound Callback (exposed to Data Power)

### Service Interface

This service exposed to Data Power to receive and process any of the five FI inquiry callback services: GetAccountsInfoCallback, GetBalanceInfoCallback, GetDepositInfoCallback, GetLiabilityInfoCallback, GetSafesInfoCallback

### Message Flow Diagram

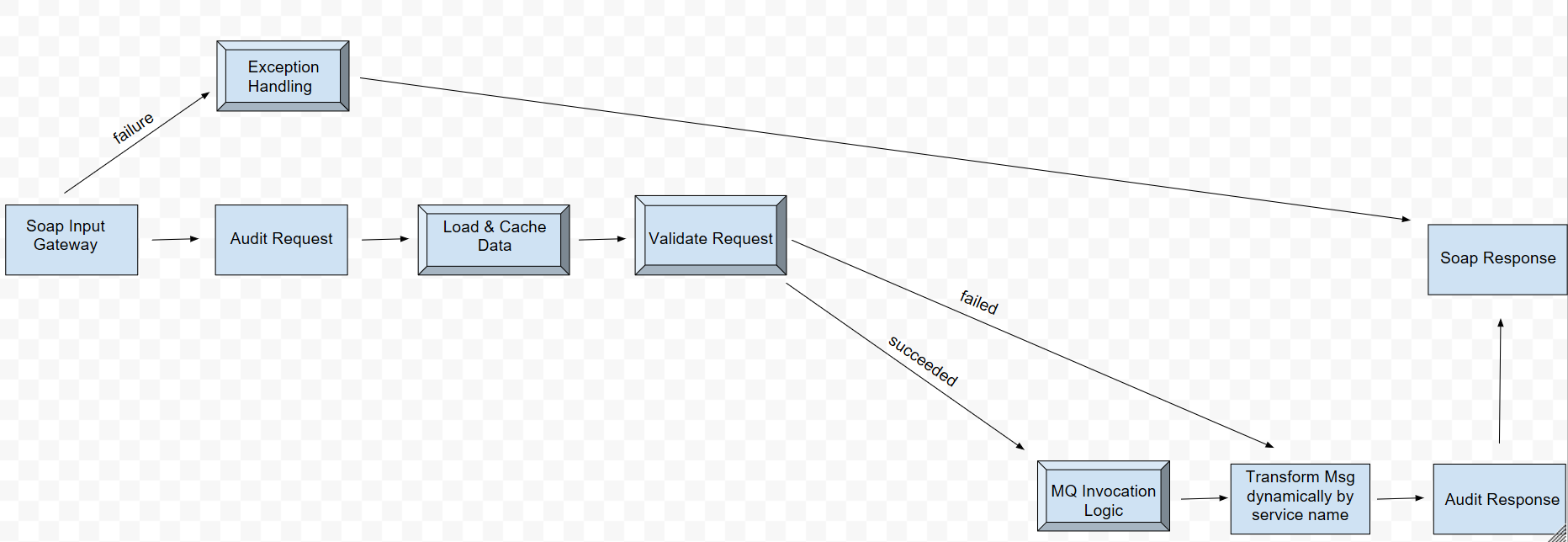


Figure 2 FI Inbound Callback Flow

* **Message Flow description**

1. Soap Input Node receives the FI callback message from Data power.
2. Audit incoming message.
3. Load and cache data sub flow. Refer to section 4.9.1
4. Validate request sub flow
   1. Check the partner Id in LOOKUP\_VALUES table then check the message duplication by Message UId and partner Id in MESSAGE\_LOG table if no duplication the request will be inserted into the MESSAGE\_LOG table. Make dependency validation such as Requester Geo Location which will be mandatory in case if the request coming from RP side (Partner Id is one of FI codes).
   2. If validation Succeeded, MQ Invocation Logic sub flow will be called.
   3. If validation failed, the message will be transformed dynamically to failed soap response. Specific esql mapping file will be called at run time using the service name (root element name of the soap message body).
5. MQ Invocation Logic sub flow. Refer to section 4.9.3
6. Generate FI soap response dynamically. Specific esql mapping file will be called at run time using the service name. Response will have status code, CRN (from Message UID of the RP request).
7. Audit outgoing response.
8. Send the soap response
9. In case of any technical error (i.e. Webservice timeout, Database or WSRR connection error, etc.), exception will be captured, propagated to first node then sent to Exception Handling sub flow
10. Exception Handling sub flow. Refer to section 4.9.2

### Request & Response Mapping

Request and response mapping will be performed dynamically by calling the corresponding esql mapping file based on the service name.

## BPM Outbound Gateway (Exposed Internally)

### Service Interface

The service exposed internally to BPM to receive and process any of the five FI inquiry request services: GetAccountsInfo , GetBalanceInfo, GetDepositInfo, GetLiabilityInfo, GetSafesInfo or any of the five RP inquiry callback services: GetAccountsInfoCallback , GetBalanceInfoCallback, GetDepositInfoCallback, GetLiabilityInfoCallback, GetSafesInfoCallback

### Message Flow Diagram

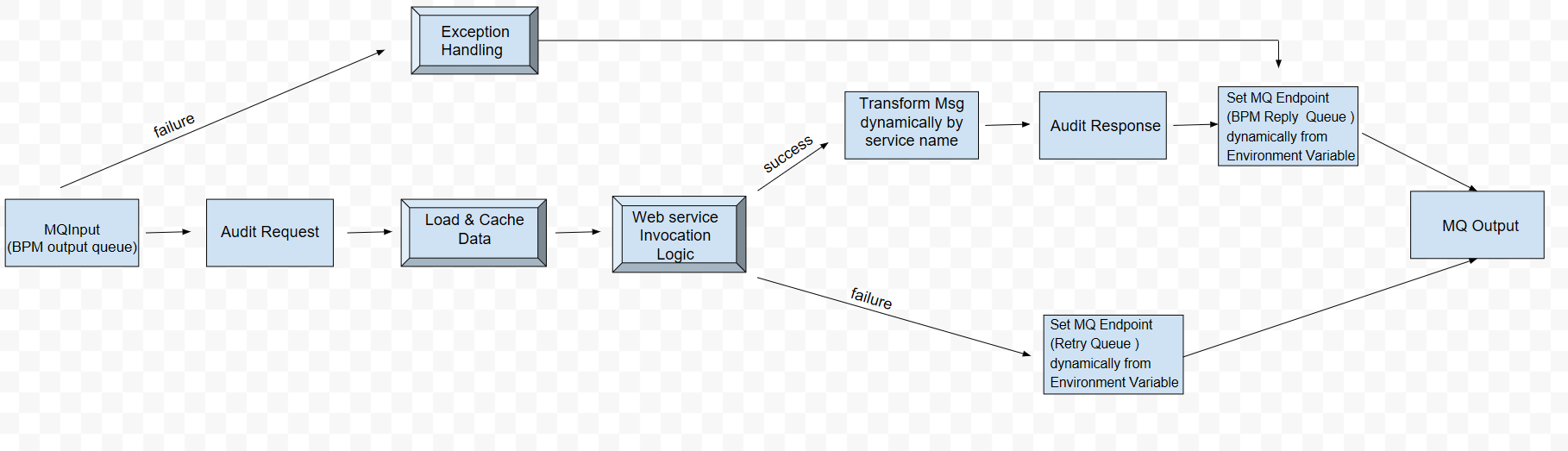


Figure 3 BPM Inbound Gateway Flow

* **Message Flow description**

1. MQ Input Node gets either request or callback message from BPM output queue.
2. Audit incoming message.
3. Load and cache data sub flow. Refer to section 4.9.1
4. Webservice Invocation Logic sub flow. Refer to section 4.9.4
5. In case of success webservice invocation
   1. Transform soap response message to either BPM FI acknowledgment message or BPM RP acknowledgment message dynamically. Specific esql mapping file will be called at run time using the service name.
   2. Audit outgoing response.
   3. Set the corresponding MQ end point details (BPM reply queue) dynamically from environment variables (populated from cache) by service name.
   4. Put the message to the BPM reply queue.
6. In case of failure webservice invocation
   1. Set the corresponding MQ end point details (BPM retry queue) dynamically from environment variables (populated from cache) by service name.
   2. Put the message to the BPM retry queue.
7. In case of any technical error (i.e. Database or WSRR connection error, etc.), exception will be captured, propagated to first node then sent to Exception Handling sub flow
8. Exception Handling sub flow. Refer to section 4.9.2

### Request & Response Mapping

Request and response mapping will be performed dynamically by calling the corresponding esql mapping file based on the service name.

## Supplementary sub flows

### Load & Cache Data Sub flow Diagram

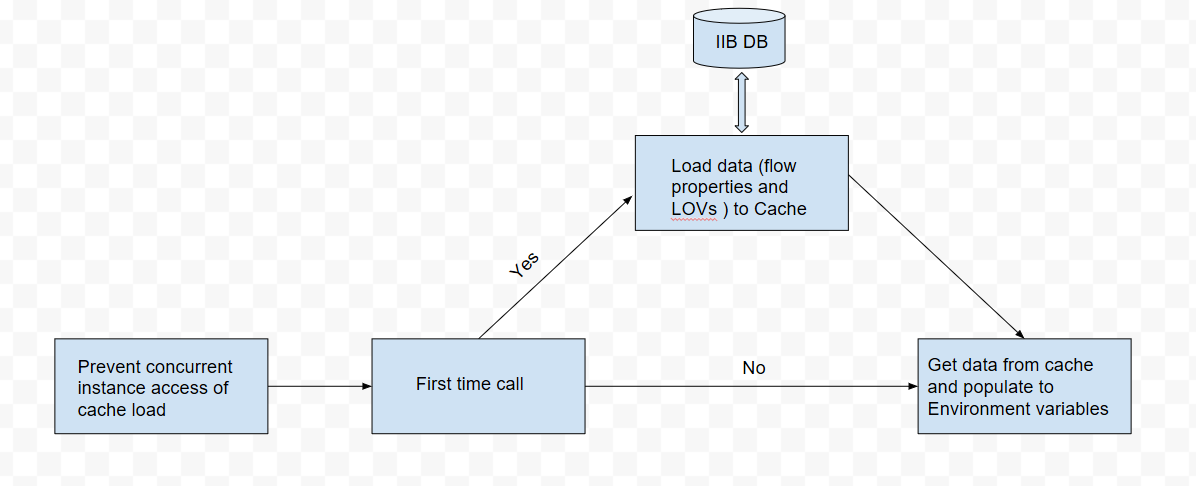


Figure 4 Load & Cache Data Sub flow

* **Sub Flow description**
  + 1. Handle concurrent instance access of cache load by updating shared variable in atomic block to check whether it is first time call or not.
    2. If it first time call, cache will be invalidated and refreshed. Data (flow properties and LOVs) will be read from IIB Database and loaded to cache. Data will be returned from cache using flow key and LOV keys then they will be populated to environment variables.
    3. If it is second time call, data will be returned from cache using flow key and LOV keys then they will be populated to environment variables.

### Exception Handling Sub flow Diagram

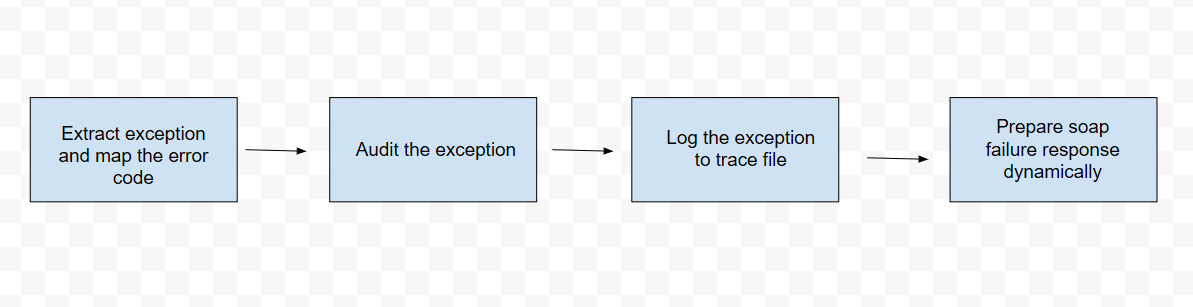


Figure 5 Exception Handling Sub flow

* **Sub Flow description**
  1. Extract the exact exception and map to the error to the corresponding error code.
  2. Audit the exception details
  3. Log the exception to trace file
  4. Prepare the failure response.

### MQ Invocation Logic Sub flow Diagram

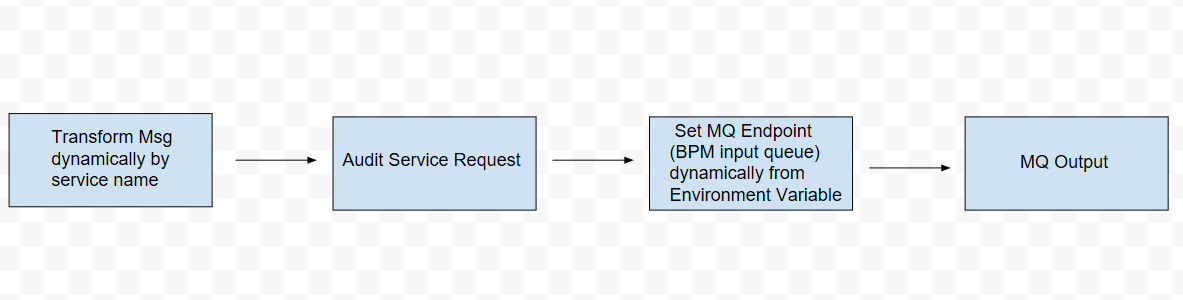


Figure 6 MQ Invocation Logic Sub flow

* **Sub Flow description**
  1. Transform incoming message request to BPM message request dynamically. Specific esql mapping file will be called at run time using the service name.
  2. Audit service request.
  3. Set the corresponding MQ end point details (BPM input queue) dynamically from environment variable (populated from cache) by service name.
  4. Put the message to the BPM input queue.

### Webservice Invocation Logic Sub flow Diagram

### 

Figure 7 Webservice Invocation Logic Sub flow

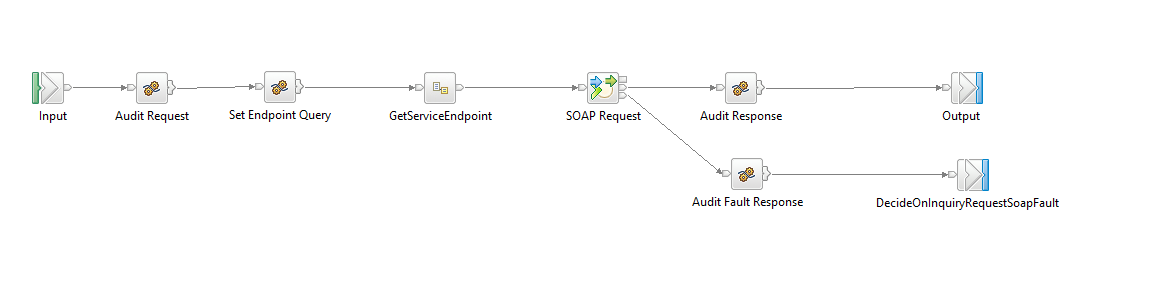
* **Sub Flow description**
  1. Transform incoming message to either FI request soap message, RP callback soap message or Portal integration request soap message dynamically. Specific esql mapping file will be called at run time using the service name.
  2. Audit service message.
  3. Set WSRR input parameters (service name, service name space, service version) dynamically from environment variables (populated from cache) by service name.
  4. Call WSRR to check if the caller (BPM) is authorized to consume the requested service then look up and return either Data Power end point URL or Portal integration end point URL based on the passed input parameters.
  5. Send the soap message to the end point URL
  6. If the webservice invocation succeeded and soap response message returned, then audit service message and route the message to success output node
  7. If the webservice invocation failed and soap fault returned, then route the message to failure output node.

## Call ODM Rule Service

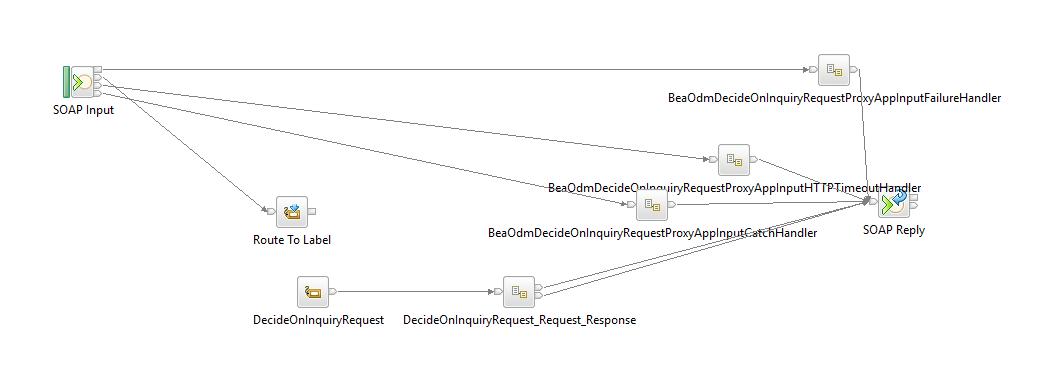
### Service Interface

This service exposed internally to receive a request from BPEL to call ODM

***Service Name:*** *BeaOdmDecideOnInquiryRequestProxyApp*



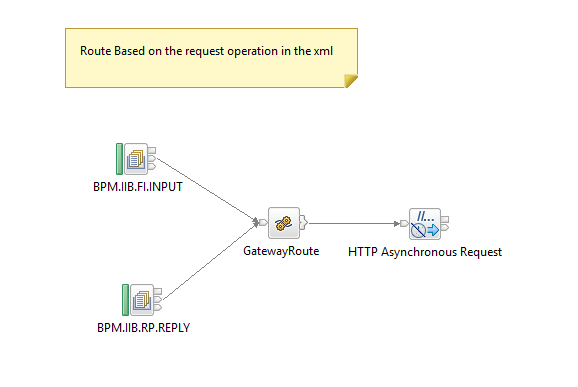
* + - **Service Flow description**
* IIB audits given request
* IIB calls WSRR and makes query to find out the service endpoint (URL from ODM service)
* IIB calls ODM to executes targeted business rules
* IIB wait the response from ODM and reply BPEL



* + - **Service Exceptional Flow**
* If there’s any failure while processing the message, IIB will use create audit message.
* Implemented in Failure handler, HTTPTimeoutHandler, InputCatchHandler sub flows which capture the exception list and prepare a soap response.
* If there’s schema validation error the message will rejected with a soap response.

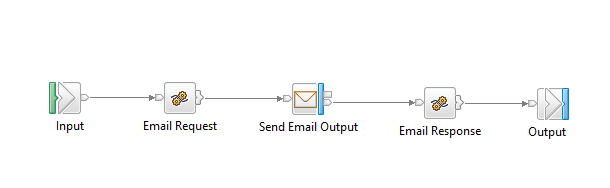
Gateway Service

This Service is used when single MQ is used between IIB and BPM Services to route messages from MQ to all services. The IIB services which use the gateway Flow will start with HTTP Input Node.



## Email Notification

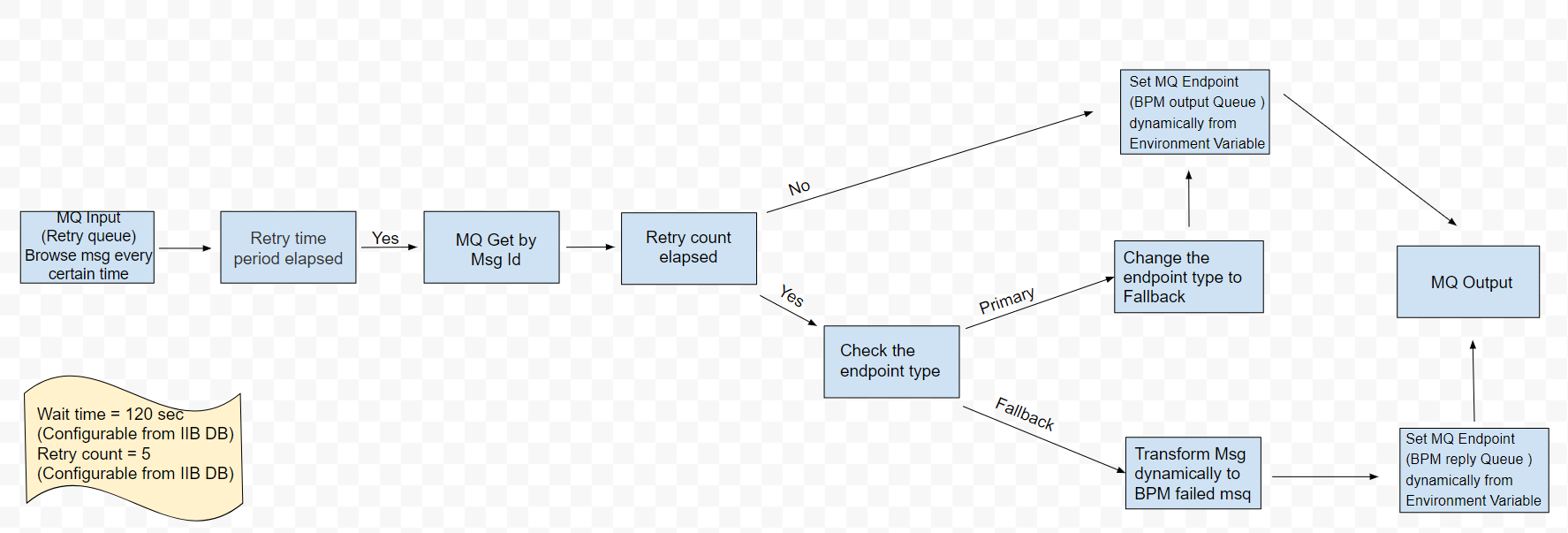
You can configure the Email Output node to send an email, with or without a single attachment, with a static subject and static text, to a static list of recipients. Using this we can send notification alerts to end consumers regarding the exception and acknowledgements by configuring SMTP Server and mentioning their emails groups in the To list.



## Retry Mechanism

This Flow will retry for backend failures, we will have current count as zero and service retry count as three, we will retry the request for backend failures and wait for 120 secs for each retry.

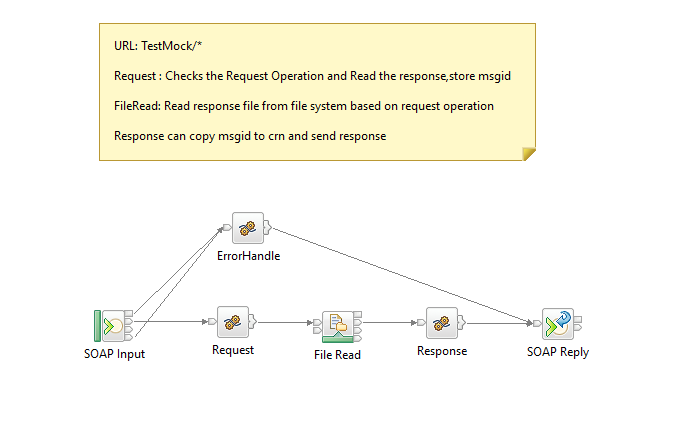
The service retry count and wait time will be configured to change the values.



* **Sub Flow description**

1. MQ Input listens to the retry queue and browse the messages in the queue every N seconds.
2. Compare the current time to the message creation time. If the retry time period has elapsed, then read the message from retry queue by Msg Id. If the retry time period has not elapsed, then do nothing.
   1. If the retry count has not elapsed, then send the message to the BPM output queue to retry the message.
3. If the retry count has elapsed, then check the endpoint type
4. If the endpoint type is Primary, then change the endpoint type to Fallback. Put the message to the BPM output queue to retry the new message (FI Portal message).
5. If the endpoint type is Fallback, then transform the message to BPM FI failed response dynamically. Specific esql mapping file will be called at run time using the service name. Put the message to the BPM reply queue to send failure acknowledgment to BPM.
6. Set the corresponding MQ endpoint details dynamically from environment variable (populated from cache) by service name.
7. Put the message to the specified queue.

## Mock Flow for Soap

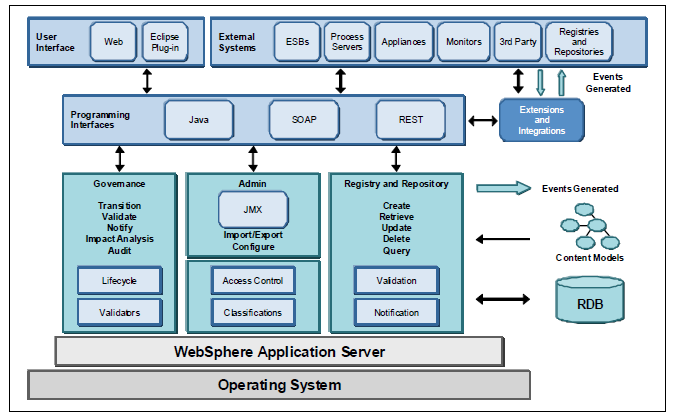


**General Note the following general rules for Data Power flow:**

1. The Data power will validate the request against the schema validation.
2. The DataPower will fetch the external systems URL from the WSRR for the PR and FI
3. If request comes to the Data power for a specific operation from an external entity, the DataPower will check from the WSRR to check if it has the privileges to access this operation or not before sending it to the IIB.
4. The IIB will make the schema validation based on the channel which send the request (ex. If portal the schema validation will be on otherwise it will be switched off)
5. All the fields dependency validation will be done in the IIB layer.

# WSRR Services Design

The below diagram depicts key components in WSRR architecture.



WSRR Architecture

The WebSphere Service Registry and Repository plays a major role in all four phases of the SOA lifecycle (Model, Assemble, Deploy and Manage).

**It helps in:**

**Manage Service Metadata:**

During service modelling, use the WSRR to create or re-use service descriptions, taxonomies, XML Schemas and other service metadata artifacts.

**Find a Service:**

During service development or assembly, use the WSRR to locate services for reuse and enable the composition of new composite applications from existing services.

**Publish a Service:**

During service deployment, publish service descriptions to the WSRR. You can augment the service descriptions generated by other tools, promoting reuse.

**Enable SOA Governance:**

The WebSphere Service Registry and Repository is the core that underpins governance in the SOA Foundation, to help you manage your overall SOA infrastructure.

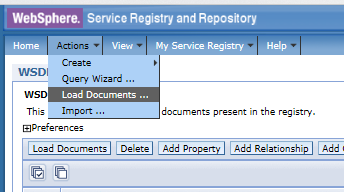
**Facilitate Service Interaction:**

During service execution use the WSRR to access service metadata and endpoint information as well as capture service metrics key so you can assess key performance indicators against your business and operational objectives.

## MQ Service Endpoint

MQ Services WSLDs are imported into WSRR.

To load WSDL document go to page:



and select “Load Documents“action. Linked XSDs are shown and for missing XSDs there is link to load them.

For MQ Services, additional objects should be created manually and linked with automatically created objects on WSDL import action. See details in MQ Services WSRR model chapter.

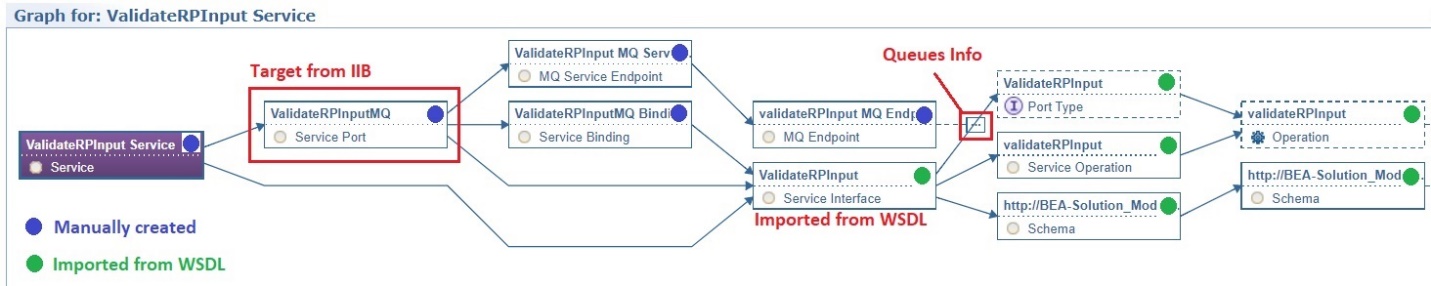
MQ Services created in WSRR are:

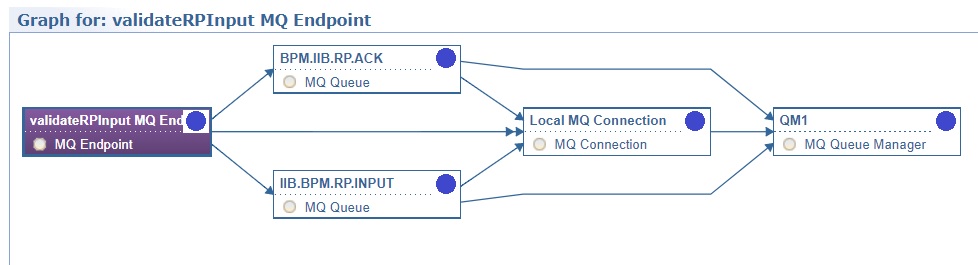
- GIInputDispatcher Service

- GIReplyDispatcher Service

- ValidateRPInputMQ Service

In the picture below, it is shown which elements of MQ Service Model are created manually and which are imported automatically and how they are connected. Only WSRR predefined relations are used to relate objects in model.





Manually created objects are:

* Service
* Service Port
* MQ Service Endpoint
* Service Binding
* MQ Endpoint
* MQ Queue(s)
* MQ Connection
* MQ Queue Manager

Integration points of manually created objects with automatically created objects are next relations:

* Service to Service Interface
* Service Binding to Service Interface

## SOAP Service Endpoint

SOAP Services Models are created automatically by importing WSDL document. There are only custom objects to support routing to different partners, but that concept is explained in next chapter “Routing“.

SOAP Services created in WSRR are:

- DecideOnBankResponseDecisionService

- DecideOnInquiryRequestDecisionService

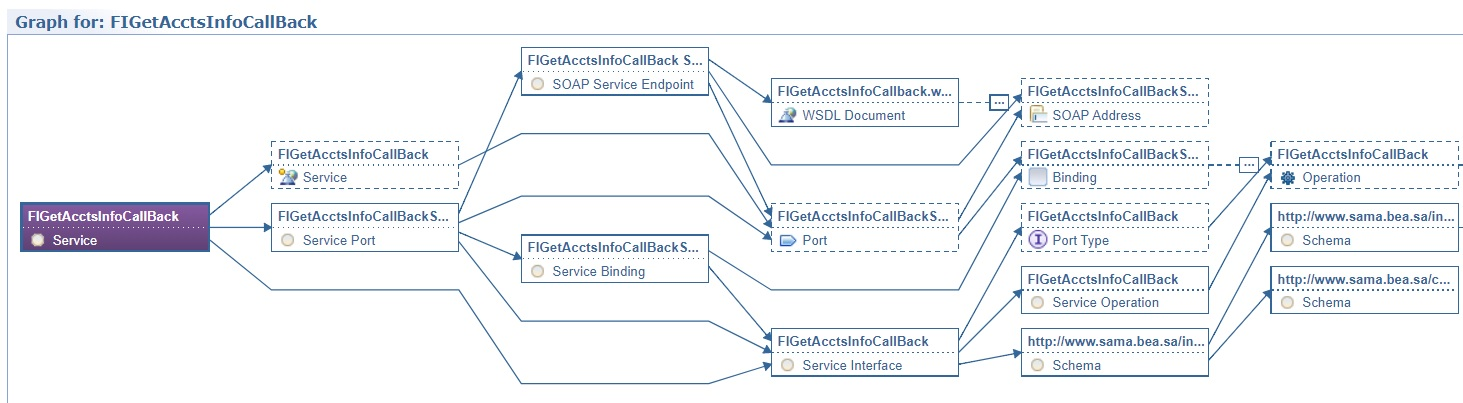
- FIGetAcctsInfo

- FIGetAcctsInfoCallBack

- RPGetAcctsInfo

- RPGetAcctsInfoCallback

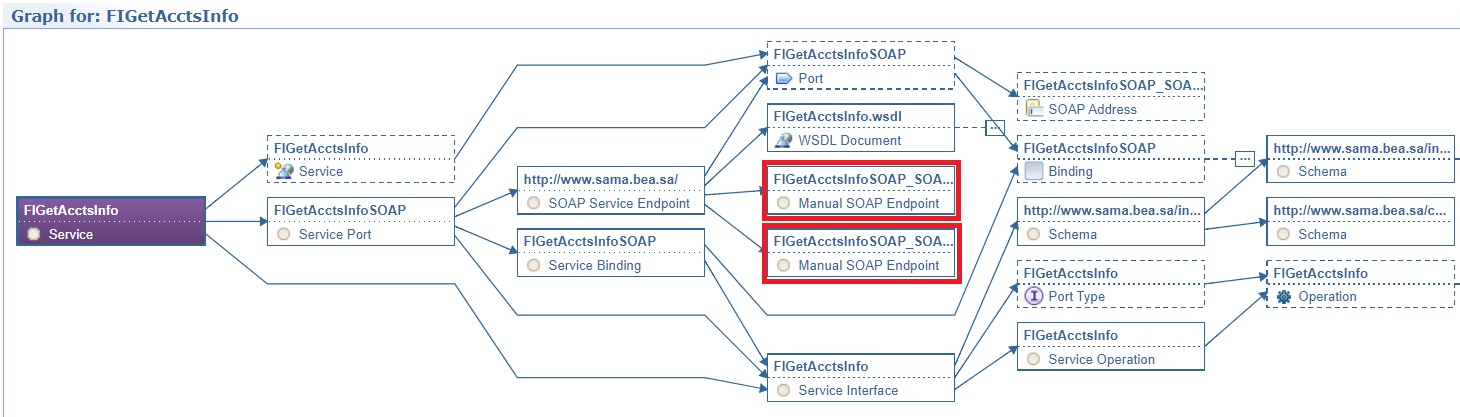
In the picture below, there is example of SOAP service model, automatically created by WSDL import.



## SOAP Service Model to Support Multiple Endpoint Routing

For service FIGetAcctsInfo, routing to different partners must be implemented, based on partner id. Each partner has its own service endpoint.

Automatically imported service model is extended with manually created SOAP Endpoints marked with red in the picture below.

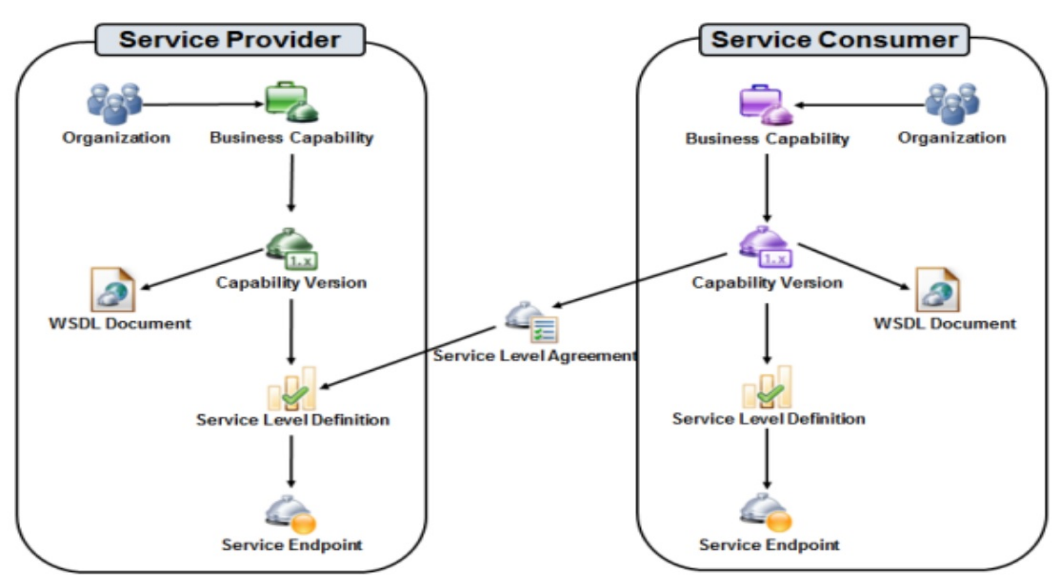


For each Manual SOAP Endpoint, SOAP Address is created which is container for service location. Each SOAP address has additional custom parameter called “partnerId“.

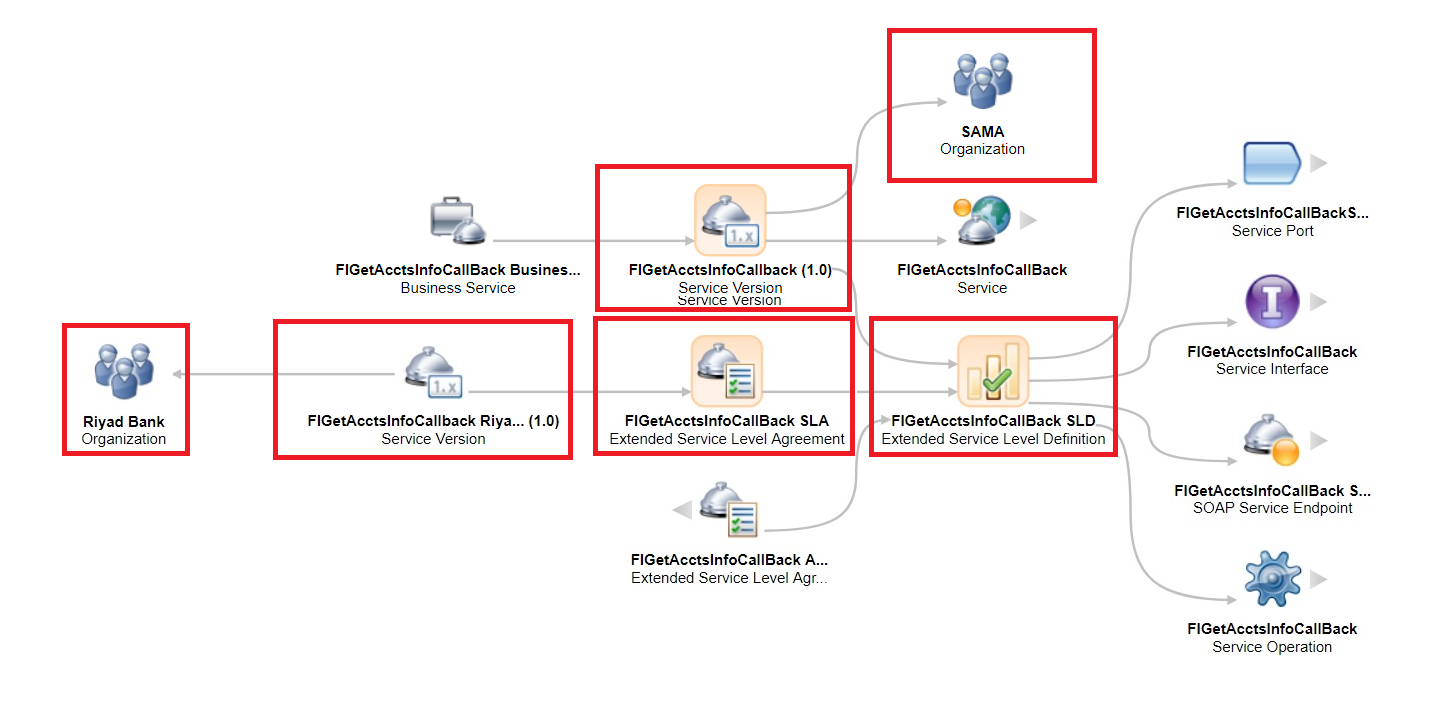
## Authorization

Authorization is supported by introducing Service Level Definitions (SLDs) and Service Level Agreements(SLAs) to link capabilities of Service Provider and Service Consumer. DataPower should be able to subscribe to Service and automatically enforce Attached Policies from WSRR. If there is Service Level Agreement between capability of Service Provider and Service Consumer, it means that Service Consumer is authorized to call Service. Additional Anonymous SLA with Attached Policy “RejectAll“is related with SLD, which will block all unauthorized access to Service.

High Level Authorization Model is shown in the picture below.



In this example shown in the picture below, provider organization is SAMA, which provides Service Version FIGetAcctsInfoCallback. There is SLD for that Service Version. Service Level Agreement connects SLD with FIGetAcctsInfoCallback Riyad Bank Service Version which represents Riyad Bank capability to call exposed service.



Service Level Definition defines by XPath where in the message should be stored Consumer ID used to identify service consumer.

DataPower should subscribe to Service Version. All attached policies should be automatically applied. If model is designed as described above, DataPower should allow only calls from Consumers with SLA between its capability version and SLD of provider.

TO BE DEFINED: What should be used as Consumer ID? DataPower could extract client id from certificate, put that client id temporarily in SOAP message header, enforce attached policies by subscription to Service Version and remove client id from SOAP header before forwarding message.

## WSRR Installation

Please refer Services Specs\Development\WSRR\_Installation.doc

## Integration with WSRR to IIB

Please refer Services Specs\Development\WSRR\_Integration.doc

## Integration with WSRR to Datapower

Please refer Services Specs\Development\WSRR\_Integration.doc

## Promotion

The process of copying an entity from one WSRR instance to another is called promotion, and the promotion feature performs the copy operation automatically when the entity undergoes an appropriate lifecycle transition as a result of your governance activity.

The promotion type. This can be any of the following types:

**synchronous**: a WSRR entity is promoted immediately after it undergoes an appropriate lifecycle transition.

**manual**: all of the promotion data is stored in a .zip file, which then needs to be manually loaded into the target WSRR instance. You can import the promotion data into the target system later by using the WSRR web UI, or you can write a script that performs the import. You can then schedule the script to run at a particular time.

**asynchronous**: you schedule the promotion to occur at a later time.