

InComm Holdings Inc.

Vanilla Reload Scan

**Technical Design**

Version 1.0

****

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** |
| **Name** |  |  |  |
| **Role** |  |  |  |
| **Signature** |  |  |  |
| **Date** |  |  |  |

Table of Contents

[1.0 Executive Summary 3](#_Toc380525363)

[1.1 Purpose and Audience 3](#_Toc380525364)

[1.2 Document Scope 3](#_Toc380525365)

[1.3 References 3](#_Toc380525366)

[1.4 Definitions, Acronyms and Abbreviations 3](#_Toc380525367)

[2.0 Design Overview 3](#_Toc380525368)

[2.1 Design Objectives 3](#_Toc380525369)

[2.2 Assumptions and Dependencies 3](#_Toc380525370)

[2.3 Rules of Thumb 3](#_Toc380525371)

[3.0 Service Design 3](#_Toc380525372)

[3.1 Service Overview 3](#_Toc380525373)

[3.2 Operations Overview 3](#_Toc380525374)

[4.0 <Service Name>\_<Service Version> - <Operation Name>\_<Operation Version> (One SECTION for each Operation) 4](#_Toc380525375)

[4.1 Overview 4](#_Toc380525376)

[4.2 Process Flow 4](#_Toc380525377)

[4.3 Technical Components 5](#_Toc380525378)

[4.3.1 BW Components 5](#_Toc380525379)

[4.3.2 Schemas 6](#_Toc380525380)

[4.3.3 Translation / Transformation 6](#_Toc380525381)

[4.3.4 Audit and Logging 6](#_Toc380525382)

[4.3.5 Exception Handling 6](#_Toc380525383)

[4.4 Testing Requirements 7](#_Toc380525384)

[4.4.1 Unit Testing 7](#_Toc380525385)

[4.4.2 Functional Testing 7](#_Toc380525386)

[5.0 Appendix A – Change Log 8](#_Toc380525387)

# Executive Summary

## Purpose and Audience

This document provides detailed high level technical design information for TIBCO API implementation of VRS project.

This document is intended for InComm Technical team (Architects to review , Support personnel etc)

## Document Scope

Detailed High level design

## References

Use this section to fill-in reference documents (standards, policies, best practices etc) used

|  |  |
| --- | --- |
| ***Document#*** | ***Document Title*** |
|  |  |
|  |  |

## Definitions, Acronyms and Abbreviations

Use this section to fill-in definitions of acronyms & abbreviations used in this document

|  |  |
| --- | --- |
| ***Term, Acronym*** | ***Definition*** |
|  |  |
|  |  |

# Design Overview

## Design Objectives

## Assumptions and Dependencies

## Rules of Thumb

# Service Design

## Service Overview

|  |  |  |
| --- | --- | --- |
| ***Service Name*** | ***Service URL*** | ***Service Description*** |
| EnableDisableCardAndClearDevices | /services/vrs/cards/action | Enable Card : This API is meant for enabling a previously disabled card  Disable Card : This API is meant for disabling an active card  Clear Devices associated with Card : This API is meant for clearing device identifiers associated with a card, intended for use by customer care personnel |
| RequestReload | /transactions/vrs | The Request Reload API is used to request a reload using the Vanilla Reload Scan system |
| RetrieveVRSMerchantList | /services/vrs/merchants?merchantFilter={filterstring} | This API is used to retrieve the list of merchants participating in the VRS program |
| ValidateToken | /services/fstoken | This API is meant to validate if the token passed in as part of the API request corresponds to a valid FS token |

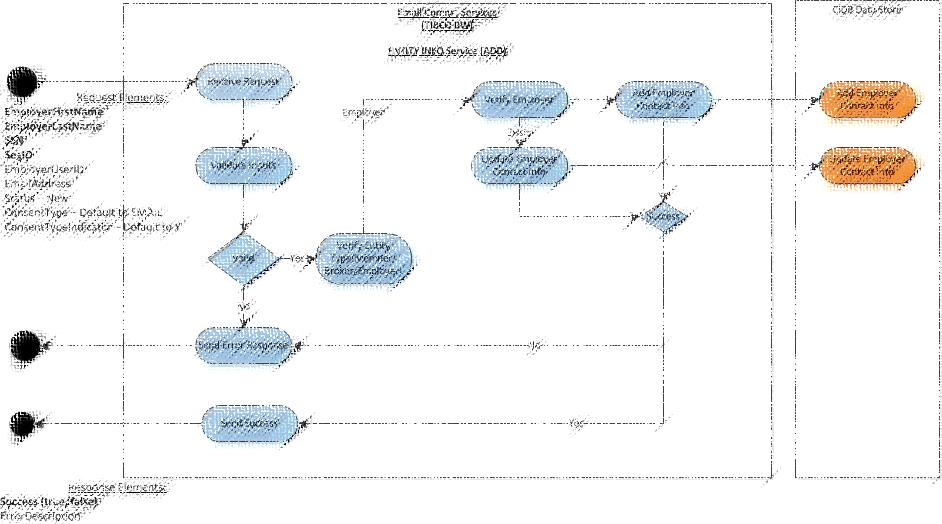
## Operations Overview

|  |  |  |  |
| --- | --- | --- | --- |
| ***Service Name*** | ***HTTP Action*** | ***Operation*** | ***Description*** |
| EnableDisableCardAndClearDevices | PUT | ENABLE | Enable VRS Card |
| DISABLE | Disable VRS Card |
| POST | CLEAR\_DEVICES | Clear Devices associated with Card |
| RequestReload | POST | Request Reload | Request Reload of VRS card |
| RetrieveVRSMerchantList | GET | RetrieveVRSMerchantList | Retrieve list of VRS merchants |
| ValidateToken | PUT | ValidateToken | Validates if the token passed in as part of the API request corresponds to a valid FS token |

# <Service Name>\_<Service Version> - <Operation Name>\_<Operation Version> (One SECTION for each Operation)

## Overview

## Process Flow

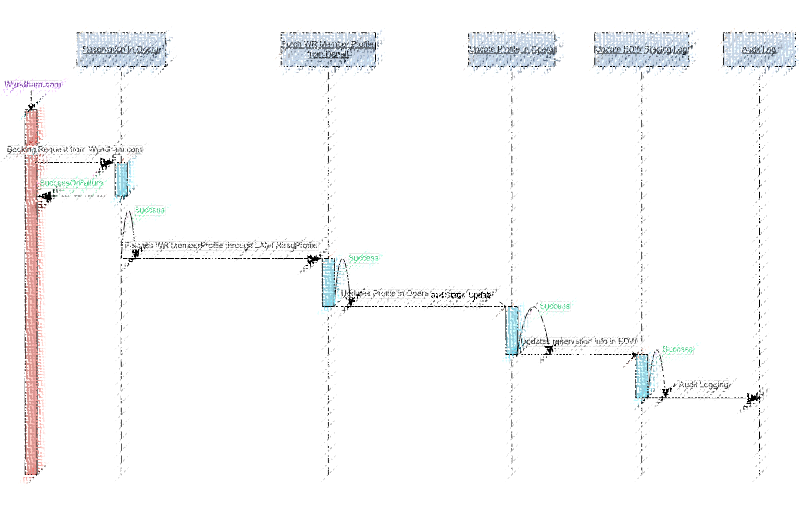


SAMPLE

PROCESS FLOW

***Fig: Sample Process Flow***

<<Use a process flow diagram, like above, to depict the operation implementation>>

****

SAMPLE

SEQUENCE DIAGRAM

***Fig: Sample Sequence Diagram***

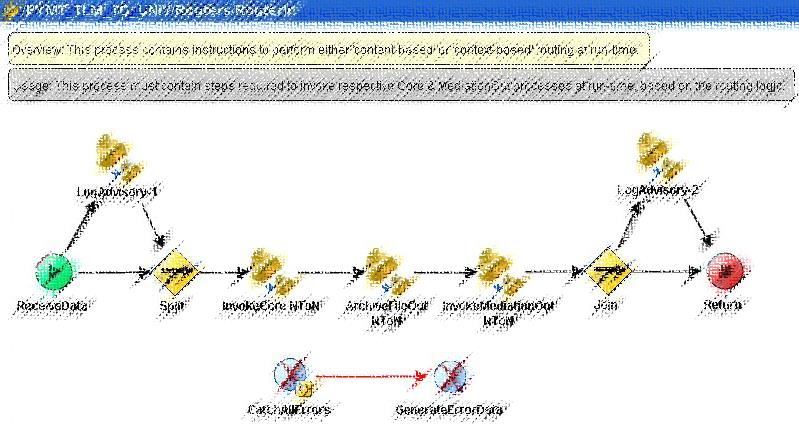
|  |  |
| --- | --- |
| Sequence# | Sequence Detail |
| 1 | <Detail the step here> |
| 2 |  |
|  |  |

## Technical Components

### BW Components

#### 4.3.1.1 <Process Name>.process (One SECTION for each Process)

<Describe the process in brief>



SAMPLE PROCESS

| **Activity Name** | **Activity Type** | **Description** | **Configuration** | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Value** |
| <Activity Name> | <Activity Type> | <Describe how it will be used> | Input | <Value goes here> |
| Output | <Value goes here> |
| Spawn | <Checked: Y/N?> |
|  |  |  |  |  |
|  |  |
|  |  |

### Schemas

<Include the schemas to be used in the operation implementation>

### Translation / Transformation

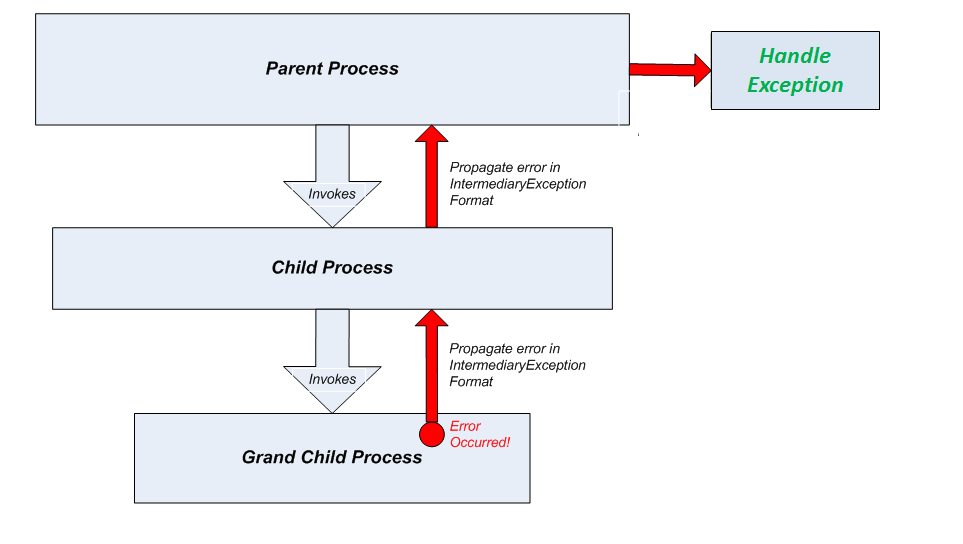
<Include the mapping sheet to be used in the operation implementation>

### Audit and Logging

<Mention the design of audit and logging that will be used in the operation implementation. Make sure this caters to the audit and logging requirements mentioned in the requirements specification document>

### Exception Handling

<Mention the design of exception handling that will be used in the operation implementation. Make sure this caters to the exception handling requirements mentioned in the requirements specification document>

******

***Fig: Exception Propagation & Handling***

Exceptions must be handled gracefully as described below.

In the above diagram, a Parent Process invokes a Child Process in-line which in turn calls a Grand Child Process. Let us assume an error has occurred in the Grand Child Process. Error will be caught in the Grand Child Process where the actual error details & stack trace are captured and routed through the exception schema to the Child Process and then on to the Parent Process. It is then handled gracefully from Parent Process.

#### 4.3.5.1 System Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
| <error> | <reason> | <handling strategy> |
|  |  |  |
|  |  |  |

#### 4.3.5.2 User-defined Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
| <error> | <reason> | <handling strategy> |
|  |  |  |
|  |  |  |

## Testing Requirements

### Unit Testing

|  |  |  |
| --- | --- | --- |
| Test Case# | Unit Test Requirement | Expected Result |
| UTC-001 | <unit test requirement> | <expected result> |
| UTC-002 |  |  |
|  |  |  |

### Functional Testing

|  |  |  |
| --- | --- | --- |
| Test Case# | Functional Test Requirement | Expected Result |
| FTC-001 | <functional test requirement> | <expected result> |
| FTC-002 |  |  |
|  |  |  |

# Appendix A – Change Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Version Number*** | ***Changes Made*** | | | |
| Vx.y | Initial baseline created on xx-Feb-2014 | | | |
| Vx.y |  | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
|  |  |  |  |
| Vx.y |  | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
|  |  |  |  |
| Vx.y |  | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
|  |  |  |  |