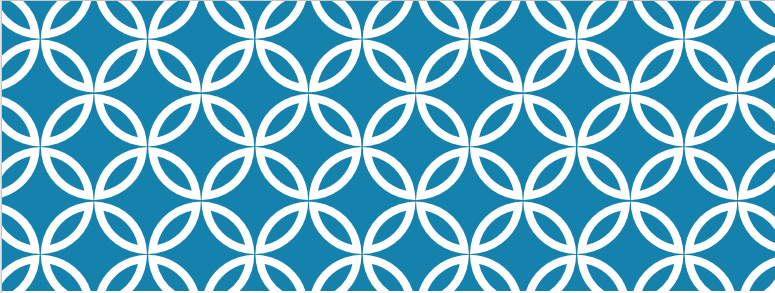
a



MOSIP

Modular Open Source Identity Platform

Version 0.1 | 24 Aug 2018

EC Application - Setup/ Update - LLD

Copyright Information

This document is the exclusive property of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; the recipient agrees that they may not copy, transmit, use or disclose the confidential and proprietary information in this document by any means without the expressed and written consent of Mindtree. By accepting a copy, the recipient agrees to adhere to these conditions to the confidentiality of \_\_\_\_\_\_\_\_\_\_\_\_\_ practices and procedures; and to use these documents solely for responding to \_\_\_\_\_\_\_\_\_\_\_\_\_\_ operations methodology.

Revision History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver** | **Change Description** | **Sections** | **Date** | **Author** | **Reviewer** |
| 0.1 | First Draft | All | 24-Aug-18 | Omsaieswar Mulakaluri |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

References

|  |  |  |  |
| --- | --- | --- | --- |
| No | Document Name | Ver. | Location |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Glossary

|  |  |  |
| --- | --- | --- |
| **Terminology** | **Definition** | **Remarks** |
| EC | Enrollment Client / ID Issuance Client application. |  |
| IDIS | ID Issuance client application. |  |
| OSGI | Container where the bundled binary jars would be running. |  |
| Tyco | It is a build tool, to build and generate the Equinox based bundles, product and update site. |  |
| Equinox | It is a library, provided by eclipse community to prepare the OSGI bundle. |  |

Table of Contents

[Copyright Information 2](#_Toc523474826)

[Revision History 2](#_Toc523474827)

[References 2](#_Toc523474828)

[Glossary 2](#_Toc523474829)

[Table of Contents 3](#_Toc523474830)

[Part A: Background 5](#_Toc523474831)

[1 Introduction 5](#_Toc523474832)

[1.1 Context 5](#_Toc523474833)

[1.2 Purpose of this document 5](#_Toc523474834)

[2 Scope 5](#_Toc523474835)

[2.1 Functional Scope 5](#_Toc523474836)

[2.2 Non Functional Scope 5](#_Toc523474837)

[2.3 Assumption 6](#_Toc523474838)

[2.4 Out of Scope 6](#_Toc523474839)

[3 Technical Approach 6](#_Toc523474840)

[3.1 Design Detail 6](#_Toc523474841)

[3.1.1 Process Flow Diagram 7](#_Toc523474842)

[3.1.2 EC Setup – Build Process: 7](#_Toc523474843)

[3.1.3 EC Setup - Product - Generation: 7](#_Toc523474844)

[3.1.4 EC Setup – Update Site Generation: 8](#_Toc523474845)

[3.2 Class Diagram 8](#_Toc523474846)

[3.3 Sequence Diagram 9](#_Toc523474847)

[4 Success / Error Code 9](#_Toc523474848)

[5 Dependency Modules 9](#_Toc523474849)

[6 Database - Tables 9](#_Toc523474850)

[7 References 10](#_Toc523474851)

[8 Pending Items 10](#_Toc523474852)

Part A: Background

# Introduction

## Context

MOSIP is developed as an open source framework project. The java standard design principles will be followed to design the component.

## Purpose of this document

This document provides the low level technical design approach of a particular functionality in MOSIP Platform. It details out the in depth technical area of a particular scope.

# Scope

## Functional Scope

* User should have access to admin portal to download the ***base setup*** of an application.
* User should be able to download the OS specific [windows/ Unix] “ID Issuance client” application from Mosip online admin portal using their respective credentials.
* User should also be able to launch the [***full setup***] application from dongle / USB device.
* User should click on the exe/ .sh file to launch the application once the application is downloaded to the local machine.
* Client machine will have online internet access, to download the latest software binary bundles from remote server [Mosip platform binary repository]. The software update should happen automatically without any manual interaction.

## Non Functional Scope

* The ID Issuance client application setup should work in both windows and unix platform.
* The application should be downloaded from the Secured based url.
* The authentication should be implemented while downloading the application from SSL based url.
* Logging of all the process along with the success or failure reason with status.
* The client application should use only the signed binary jars provided by the MOSIP source repository.
* Application should be configurable:
  + Configuration:
    - Application installable location
    - RAM, processor and hard disk requirement.
    - Audit log location.
    - Packet creation location.
    - Local db connection. [?]

## Assumption

* System should have either internet access / USB dongle access to install or run the application.
* If the application launched from dongle and write the application output into the dongle, then the dongle should have :
  + Read/Write access permission to read and write enrollment packets/ logs.
  + Enough space to create packets/ logs.
* If the application launched from dongle and write the application output into the system hard disk, then :
  + Dongle: Read permission to read the application configuration.
  + Hard disk: Should have read permission and enough space to create packets/ logs.
* Application users should take the ownership of the system base requirement validation based on the “MOSIP – implementation document”.

## Out of Scope

* System Hard Disk space, RAM and CPU processor base requirement validation during setup process.

# Technical Approach

## Design Detail

The Enrollment Client software kit design approach has been detailed out below.

### Process Flow Diagram

The below diagram depicts the ‘Enrollment Client’ software setup/ update and launch process.

Figure 1

User Login Admin Portal

Select the “Enrollment Client” Software Setup Menu

Click and download the OS specific ‘Enrollment Client’ software kit to be downloaded

Save the downloaded software to the particular location in the Local Hard disk.

Unzip the downloaded Kit

Inside Extracted folder, click ‘launch.exe’ to launch the EC application

While launching EC application, if there is any software update it will get downloaded from Mosip build repository [when user online]

Enrollment Client application gets launched.

### EC Setup – Build Process:

* Use the “Eclipse Equinox” OSGI framework’s library for implementing the bundle concepts.
* Prepare the bundles for “Enrollment client” product using OSGI library.
  + UI component bundle.
  + UI processor component bundle.
    - Embed the required dependent libraries as a jar.
* Any bundle update should be created with the newer version.
* ‘Tyco’ libraries should be used to build the entire EC application.
* Tyco - Prepare the build script to generate the initial product and update sites along with the versions.
  + Initial Product Generation – this should be generated during the first build process.
  + Update site Generation – further build should generate the updated bundle with the newer version in a specific ‘update site’ location.
* Once the [product](#_EC_Setup_-) is generated, create the zip out of it and push it into the admin portal

### EC Setup - Product - Generation:

* As an output of Equinox OSGI build, the product bundle should be created with initial setup features.
* Prepare the Initial software Kit as provided in the below structure [zipped] and share the respective downloadable link through Admin portal.
* Configure the product, with the following features:
  + The product kit should be generated for multiple OS [windows, unix].
  + Through secured mode of https [SSL] connection, the client software update should happen from Mosip build repository.
  + While downloading the application the authentication should be passed.
  + While executing the application, should verify whether all the jars are signed. If not, then throw exception:
  + **Verify the signed jar >**
    - Ref: <https://docs.oracle.com/javase/tutorial/deployment/jar/verify.html>
    - jarsigner -verify *jar-file*

### EC Setup – Update Site Generation:

* The binary of the “Enrollment client” product should be available in the standard repository which is to be accessible outside the server network through the API Gateway. This should be accessible through SSL based https url using valid credentials. This is called as ‘update’ site in OSGI terminology.
* All the binary jars [bundles] generated out of the build process should be signed
  + Create platform ‘MOSIP’ specific key [private and public] file [using alias] and maintain the same into the key repository.
  + Use the private key to sign the jar files generated out of the build process.
  + **Command to sign the jar >** jarsigner *jar-file alias*
    - jar-file is the pathname of the JAR file that's to be signed.
    - alias is the alias identifying the private key that's to be used to sign the JAR file, and the key's associated certificate.
    - Ref: <https://docs.oracle.com/javase/tutorial/deployment/jar/signing.html>
  + Use the required options provided in the ‘jarsigner’ command to sign the jar using the right private key.
* The update site URL to be configured in [‘EC Setup’ product configuration](#_EC_Setup_-) file to further download the product whenever there is an update in the site.

### EC Launch:

* Once start the application, load the application bootup class and that will intern call the
  + ApplicationConfig loader class to load the configurations.
  + Load and initiate the Spring Context.
  + System Health Checker to validate the system requirements are met before launch the application.
    - Hard Disk space, RAM and CPU processor

## Class Diagram

NA

## Sequence Diagram

NA

# Success / Error Code

|  |  |  |  |
| --- | --- | --- | --- |
| **Code** | **Type** | **When** | **Description** |
|  | Error | When launching EC Client | Unable to connect to the server to update the product. |
|  | Error | When launching EC Client | Unable to connect to the server using the provided authentication. |
|  | Error | When launching EC Client | Unable to launch the application as the jars are not signed. |
|  | Error | When launching EC Client | Unable to launch the application due to hand shake error [if the SSL certificate expired] |
|  |  |  |  |

# Dependency Modules

|  |  |  |
| --- | --- | --- |
| **Component Name** | **Module Name** | **Description** |
| Audit Manager | Kernal | To audit the process while creating the packet. |
| Exception Manager | Kernal | To prepare the user defined exception and render to the user. |
| Log | Kernal | To log the process. |
|  |  |  |

# Database - Tables

NA

# References

|  |  |
| --- | --- |
| **User Story No.** | **Reference Link** |
| MOS-14 | <https://mosipid.atlassian.net/secure/RapidBoard.jspa?rapidView=1&projectKey=MOS> |
| MOS-61 | https://mosipid.atlassian.net/secure/RapidBoard.jspa?rapidView=1&projectKey=MOS |

|  |  |
| --- | --- |
| **Other References** | **Reference Link** |
| OSGI | <https://osgi.org/> |
| Eclipse Equinox |  |
| Tyco Build |  |
|  |  |

# Pending Items

1. Integration of spring latest version [4x] with the equinox OSGI container.
2. Spring context initialization.
3. Provide walk through to build team to setup the same in CI process.
   1. Build process setup for equinox in Jenkins.
   2. Script preparation for Build process.
   3. Setup update repo.
4. During EC launch –
   1. Hard disk, storage – validation.
5. Health check >
   1. Should work in all the OS – need to be validated…