Test Strategy Document

DataStage to Matallion Migration

**Version : 1.0**

**Date : 9/29/2023**

**DOCUMENT HISTORY**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Revision Date** | **Summary of Changes** | **Updated By** | **Reviewed By** | **Approved By** |
| Draft | 9/26/2023 | BAU Review of Initial Outline and Draft |  |  |  |
| Draft | 9/27/2023 | Sponsor Review of Section 3.1.1. A/B Testing |  |  |  |
| 1.0 | 9/29/2023 | Current version |  |  |  |

Contents

[1. Introduction 4](#_Toc146819214)

[1.1 Document Purpose 4](#_Toc146819215)

[1.2 High Level Test Plan 4](#_Toc146819216)

[2. Testing Scope 5](#_Toc146819217)

[2.1 Test Types 5](#_Toc146819218)

[2.2 Performance Testing Scope 6](#_Toc146819219)

[2.3 Testing Activities: 7](#_Toc146819220)

[3. High Level Testing Methodology 9](#_Toc146819221)

[3.1 Test Methodology 9](#_Toc146819222)

[3.1.1 A/B Testing 9](#_Toc146819223)

[4. Assumptions 10](#_Toc146819224)

[4.1 Input Requirements: 10](#_Toc146819225)

[4.2 Dependencies: 10](#_Toc146819226)

# Introduction

## Document Purpose

The Test Strategy document represents the high-level approach required for testing the DataStage to Matallion Migration project. It identifies the scope, assumptions, issues, and types and levels of testing required be planned and executed during the project.

## High Level Test Plan

At an application level, as defined in the baseline inventory, each application will be tested based on requirements from the System Requirements Document (SRD), System Design (SDD) and Source to Target Mapping (STTM). The SDD documentation that is available for each application defines the Functional Requirements (FRs) and Non-Functional Requirements (NFRs) that will be tested. The STTM defines the data testing requirements that are specific to A/B testing portion of the strategy. The success criteria for testing is to achieve automated and efficient A/B testing with passing results and documented exceptions.

The development team is responsible for developing the application code and performing unit testing in the DEV environment. The DEV team will then replicate their code into the DEV-IIT environment. The testing framework and initial test case execution will take place in both the DEV and DEV-ITT environments. On failing test conditions, the QA team will create issues against the Features/User Stories or Test Plans in the ADO project board for the purpose of defect tracking.

Once the QA team completes their initial testing and all relevant defects related to the code or configuration have been resolved, the code base will be baselined and considered ready to move through the CI/CD deployment pipeline process. The release management team will support this process if the Conveyor Belt method is chosen. Alternatively, the CI/CD processes developed in Azure DevOps is allowed to perform direct deployments into test environments with the proper approvals.

# Testing Scope

## Test Types

**Unit Testing:** This is performed by developers to verify the functionality of the individual migration or conversion programs they develop. The programs are tested to verify that they

correctly ingest data from source systems, transform data, handle and report error

conditions, create Snowflake-compatible objects, and produce validation reports.

**A/B Testing:** A/B Testing is used to validate data structures and the results produced by ETL objects. This testing focuses on individual objects migrated from Netezza to Snowflake. It is performed by members of the migration project team and involves comparison of Snowflake objects to corresponding objects in Netezza.

* Netezza/Oracle SQL queries are converted to Snowflake SQL and validated using automated scripts.
* The validated SQL is then inserted into ETL programs or objects, replacing Netezza/Oracle SQL.
* Connection commands are redirected to point to the Snowflake EDW.
* A Temp table shall be created to verify that the input and output tables needed for each object are converted and available for testing.
* ETL objects are then executed in Snowflake test environments using Snowflake compatible tables, views, and SQL. Depending on file formats, the results may be extracted to Excel to facilitate the comparison. Automated validation scripts are used to compare the Snowflake ETL results to the results of corresponding objects in Netezza/Oracle. If unexpected differences are found, the migration and validation scripts are reviewed, remediated, and rerun until all mismatches are resolved.

**Integrated Testing:** This is the end-to-end testing of the entire module that is developed with other dependencies.

**System Integration Testing:** This is the testing done by quality assurance team based on

approved test cases. Here the jobs/sequences are executed as in production (production volume

of data) with all dependencies and the target data is tested against the requirements. A parallel

run is recommended for Netezza/DataStage environment with the same data used for system testing of Snowflake/Matillion, the testing team will compare the data of Snowflake to Netezza

after the job is run in both environments. The test cases and results are shared with the business users for review and approval/sign-off.

**Pre-production testing:** - The jobs here are run as in production before the results are compiled and produced to the business for a final sign-off.

**HWA Testing:** Testing includes the successful execution of HWA jobs. NTT creates runbooks and HCL codes HWA jobs based on the runbook created by NTT. NTT owns end to end validation of job runs. However, the support of HWA jobs is done by HCL.

## Performance Testing Scope

The following critical flows will be performance tested. Performance testing will compare the run time statistics for these jobs run in the current production environment with the performance of the replacement workflows in the mid-level testing environment.

**Supply:**

* FDA
  + All daily FDA flows
* EDR
  + Non claims flow
  + RX flows (includes Audit and Fact table loads)
  + Medical claims flow
  + Vision claims flow
  + Dental claims flow
  + EDR Monthly flow
  + PCP Attribution flow
  + Income flow

**Demand:**

* PBM eligibility flow
* Vision eligibility flow
* Dental eligibility flow
* Pharmacy Carveout eligibility flows
* Cost Plus Billing flow
* MHK flow

## Testing Activities:

* Testing focus is to validate the migration of ETL components from DataStage to Matallion, database objects from Netezza/Oracle to Snowflake.
* Functional test cases will be created for the Matallion replacement objects, these will initially be created as manual test cases. Matallion test cases will be converted to automated test cases by integrating ADO Test Plans with the Matallion API.
  + Service Account access to integration points (Snowflake, ADLS Container)
  + Verification of inbound files: availability, naming patterns, record count, frequency/schedule, duplicate/re-run, etc.
  + Positive/Negative scenarios for logical gate conditions
  + Record load counts and count comparisons
* Data test cases will be prepared for A/B testing. These will initially be executed manually and then automated by integrating the automated testing components into the future state core components (Matallion, Snowflake, Azure DevOps).
  + Control table that captures metadata for each run by application/pipeline.
  + Results table that has comparison values for incomplete matches that can be queried by analysts to view at an application/table level the delta records.
  + Reporting tool compatibility with both the Control table and Results table so that analysts can either query Snowflake directly or use a reporting application.
* Performance testing for comparing current state DataStage ETL runtimes with future state Matallion runtimes.
  + Baseline results for each application listed in the *Section 2.2*
  + Runtimes for future state runs captured from the Matallion admin console or from the HWA schedulers admin console.
  + Comparison reports created for different test execution runs in each environment starting with SIT2.
* Oracle/Netezza Stored Procedure conversions scope will include DDL and data migration specifically for the Oracle conversions.
  + Validate database object conversion.
  + Validate results written to target tables with A/B testing against the current state implementation.

# High Level Testing Methodology

## Test Methodology

### A/B Testing

The following figure, *Figure 1*, illustrates the scope of AB Testing. AB Testing is where a similar dataset is compared between 2 versions of the same table in different databases. The AB Testing methodology will be applied to both database tables and outbound files. *Section 2.1* of this document formally defines the term A/B Testing.

A diagram of a computer

Description automatically generated

**Figure 1**

Initial testing activities will take place in the DEV & DEV-IIT environment. The DEV & DEV-ITT environment will be where the A/B testing framework is created. Testing will be a combination of manual and automated testing. Test cases will be documented in Azure DevOps in the Test Plans section. Manual tests can be setup in ADO Test Plans with minimal configuration. Automated testing will cover the A/B testing scope and additional configurations and components will need to be built outside of ADO to support A/B test automation.

# Assumptions

## Input Requirements:

* Baseline Inventory
* Business Requirements (SRD)
* Detailed Design (SDD)
* Source to Target Mapping (STTM)

## Dependencies:

* All upstream source data dependencies are identified and inventoried during baseline inventory documentation.
* Upstream inputs, tables and files, are replicated in the DEV and DEV-IIT environments to facilitate Development and initial testing in DEV and DEV-IIT
* Replication of source files into the ADLS Storage Container is necessary for full testing of incremental data loads. Scheduling and inbound file automation will be necessary.