Western Digital

MMSLOADER Report

Requirements Specification

Revision: 1.0

Prepared by
Shahid Kochak

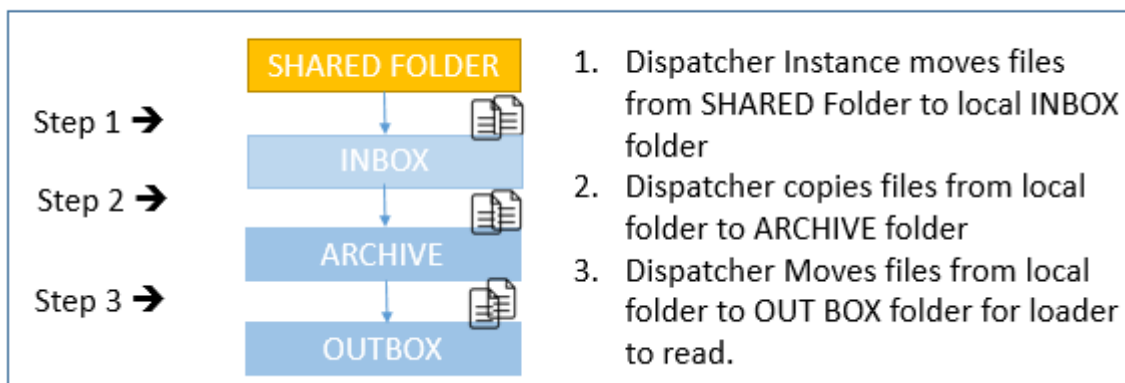**17 July 2017**

REVISION HISTORY

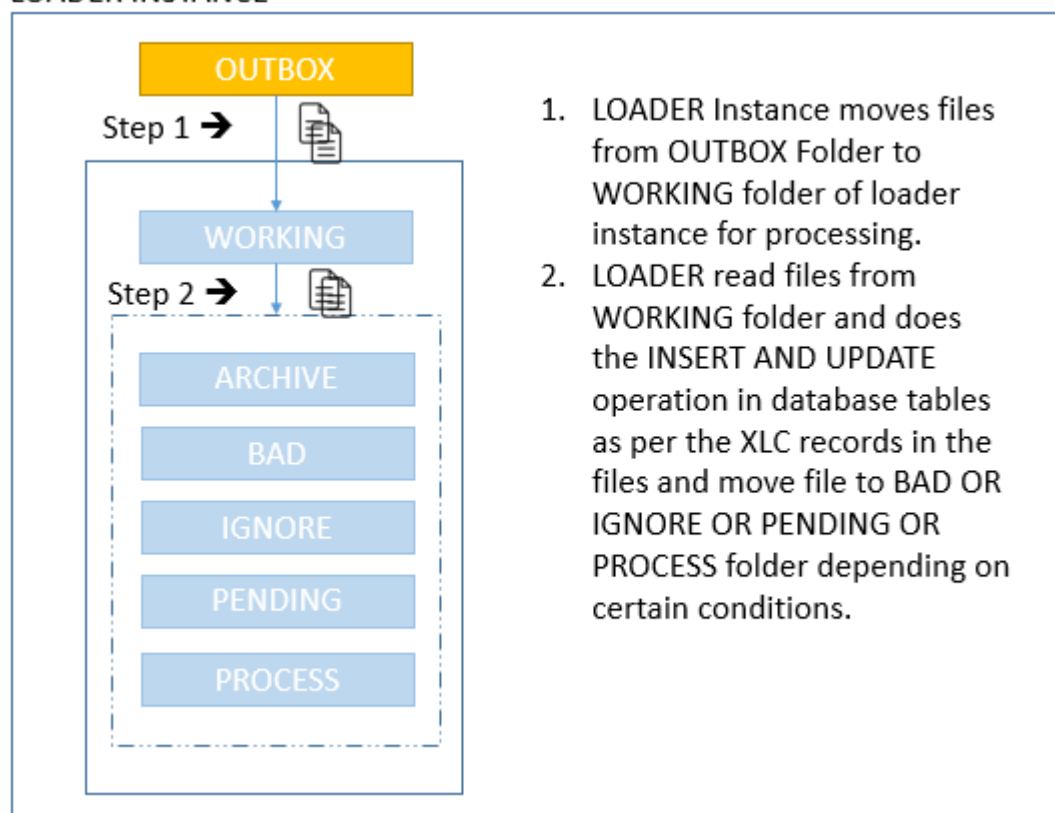| Version Number	Items/Description of change	Date Updated	Updated By
1.0	Initial Release	17 July 2017	Shahid Kochak

CONTENTS

1 OBJECTIVE

The main objective it to read **XLC RAW** files (CSV,TXT etc) and load data to respective tables depending on the configured templates.

DISPATCHER INSTANCE

Step 1 → SHARED FOLDER → INBOX → ARCHIVE → OUTBOX (Step 2, Step 3)	1. Dispatcher Instance moves files from SHARED Folder to local INBOX folder 2. Dispatcher copies files from local folder to ARCHIVE folder 3. Dispatcher Moves files from local folder to OUT BOX folder for loader to read.

LOADER INSTANCE

Step 1 → OUTBOX → WORKING → (Step 2) ARCHIVE / BAD / IGNORE / PENDING / PROCESS	1. LOADER Instance moves files from OUTBOX Folder to WORKING folder of loader instance for processing. 2. LOADER read files from WORKING folder and does the INSERT AND UPDATE operation in database tables as per the XLC records in the files and move file to BAD OR IGNORE OR PENDING OR PROCESS folder depending on certain conditions.

2 REQUIREMENTS

❑ LOADER

   a) Loader should be able to do below task

      a. Able to create folder structure per process as follows.

         i. WORKING

         ii. PENDING

         iii. PROCESS

         iv. BAD

         v. IGNORE

         vi. ARCHIVE

      b. Able to configure templates per **XLC RAW** (Csv,txt etc.) files and table.

      c. Able to read **XLC RAW** files (CSV,txt etc..) and do below operations

         i. Read files and insert or update data to respective tables as per template configuration

         ii. After reading file successfully it should move files to certain folders as per below conditions.

            1. If loader is able to read file without any issues file should be moved from **WORKING** to **PROCESS** folder.

            2. If there is some connection failure while reading a file, File should be moved from **WORKING** to **PENDING** folder.

            3. If XLC is not configured in loader database then file should be moved from **WORKING** to **IGNORE** folder.

            4. If there are some records which are **corrupt or mismatch** (**colums not matching with template columns or datatype of columns not same as defined in template**) then those records should be saved as file and moved to BAD folder.

2.1 **LOADER**

   ◼ PRE-REQUISITE

      ❑ 64 BIT OR 32 BIT WINDOWS OPERATING SYSTEM

      ❑ DOTNET FRAMEWORK 4.5 or above.

      ❑ MICROSOFT ACCESS DATABASE ENGINE **(included in Setup)**

      ❑ WD.DATACCESS DLL **(included in Setup)**

      ❑ WD.DOMAIN DLL **(included in Setup)**

      ❑ Log4net DLL **(included in Setup)**

      ❑ ORACLE MANAGED DATA ACCESS DLL **(included in Setup)**

      ❑ METRO FRAMEWORK DESIGN DLL **(included in Setup)**

      ❑ METRO FRAMEWORK DLL **(included in Setup)**

      ❑ METRO FRAMEWORK FONTS DLL **(included in Setup)**

❑ NEWTONSOFT JSON DLL **(included in Setup)**

■ TEMPLATE CONFIGURATION

    ❑ Testing environment is having below credentials.

        ■ **Server Name** : 172.21.12.73\INPXMLD2MY

        ■ **DBName** : MMSDB

        ■ **User Id** : XXXX

        ■ **Password** :XXXXX

    ❑ XLC RAW file sample Data

    In below table each column header is represented as F1,F2…FN

6/19/2017 8:08	6/17/2017 3:08	1/1/1970 8:00									………..
XSU01				EM6A1	NEPTUNE	13	58	0	WXR1A478PZ0T	10.0.49.0	
6/19/2017 8:09	6/17/2017 3:08										………..
XSCC01				EM6A1	NEPTUNE	13	58	0	WXR1A478PZ0T	XCFGS2	

    ❑ To configure templates for XLC Raw file(s) we have below table structure and JSON format for storing MAPPING information per XLC.

        ■ TABLE STRUCTURE

            ⊟ 📰 dbo.APPCONFIG

                ⊟ 📁 Columns

                      📄 RECID (varchar(2000), not null)

                      📄 CONFIG (varchar(max), null)

                      📄 CREATEDON (date, not null)

                      📄 CREATEDBY (varchar(2000), not null)

                      📄 UPDATEDON (date, null)

                      📄 UPDATEDBY (varchar(2000), null)

                      📄 ID (varchar(2000), null)

                      📄 ISACTIVE (numeric(1,0), null)

        ■ MAPPING JSON

            Example

```
{
  "Columns": [
    {
      "Index": 1,
      "ColumnName": "F1",
      "DataType": "string",
      "DefaultValue": "",
      "FieldName": "REC_ID",
      "Format": "",
      "Length": "8",
      "IsPrimary": true,
      "iLength": 8
    },
    {
      "Index": 2,
      "ColumnName": "F2",
      "DataType": "DateTime",
```

```
              "DefaultValue": "",
              "FieldName": "TIMESTAMP",
              "Format": "",
              "Length": "",
              "IsPrimary": true,
              "iLength": 0
          },
            ],
        "DbConfig": {
          "DbInstance": "abc",
          "ConnectionString":
      "Database=abc;Server=abc;User
      Id=abc;Password=abc;Connect Timeout=30;",
          "ServerName": "abc",
          "UserId": "abc",
          "Password": "abc",
          "DbProvider": "Oracle",
          "TargetTableName": "X_AUTOMATION_DOWNTIME"
        },
        "Schema": null

                            }
```

- ■ SCREEN SHOTS TO SHOW UI FOR TEMPLATE CREATION, MAPPING AND EXECUTION.

  - ■ TEMPLATE CREATION UPDATION AND LIST VIEW

In below screen you can create / edit update template and view list of templates created. (More information in User Guide)
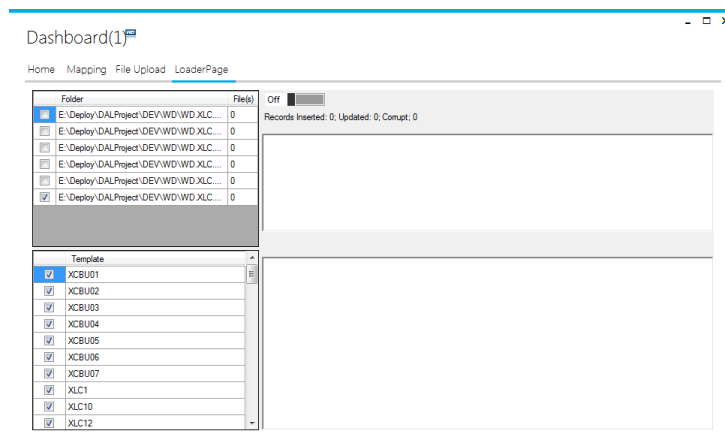

  - ■ TEMPLATE MAPPING

In below screen left side of screen shows list of templates and right side shows mapping attributes as per template selected.(More information in User Guide)
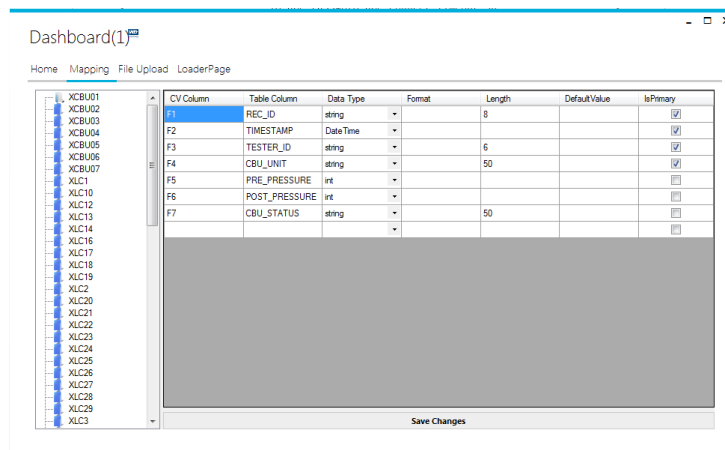
■ LOADER EXECUTAION SCREEN

In below screen left side of screen shows list of folders and list of templates.You need to select template and folders and press toggle on to process files. (More information in User Guide)

❑ **AppConfig table** is main table where we store XLC template configurations.

❑ **RecID** column stores XLC template RecIDs eg XLC53,XLC54

❑ **Config** column stores the mapping data information as json string.

   Mapping information details contains below information

   o Table information for Template where data will be inserted or updated.(`TargetTableName`)

   o Mandory **(CV COLUMN AND TABLE COLUMN)** Columns mappings of **XLC RAW** file columns with table columns.(`ColumnName, FieldName`)

   o Madatory **Data type** as per database table column.(`DataType`)

   o Optional **Length** of column as per database table column.(`Length`)

   o Optional **Data Format** of column as per database table column.(`Format`)

      Eg (DateTime : YYMMDD or yyyy-MM-dd).

   o Optional **Default Value** as per database table column.(`DefaultValue`)

   o Unique or **Primary** information as **Boolean.(`IsPrimary`)**

❑ **CreatedOn** column stores date time, when Template has been created.

❑ **CreatedBy** column stores user id.

- ❑ **UpdatedOn** column stores date time, when Template get updated
- ❑ **UpdatedBy** column stores user id.
- ❑ **IsActive** column reperesents active templates.
- ❑ **ID** column is unique identifier for AppConfig table.

2.2   Conditions and Rules:

The following are the filter conditions for data mapping and manipulation :
    a)   Create Templates for XLCs.
    b)   Map columns for XLC with Prefex as F and Suffix interger e.g F1,F2
        Note
        F1 with Table Column as Rec_ID is mandatory.
    c)   Start Loader for executing files.Process will decide whether to process files Parallel or Sequential, depending on **No. of files** to be processed and on the number of processors on the current machine.
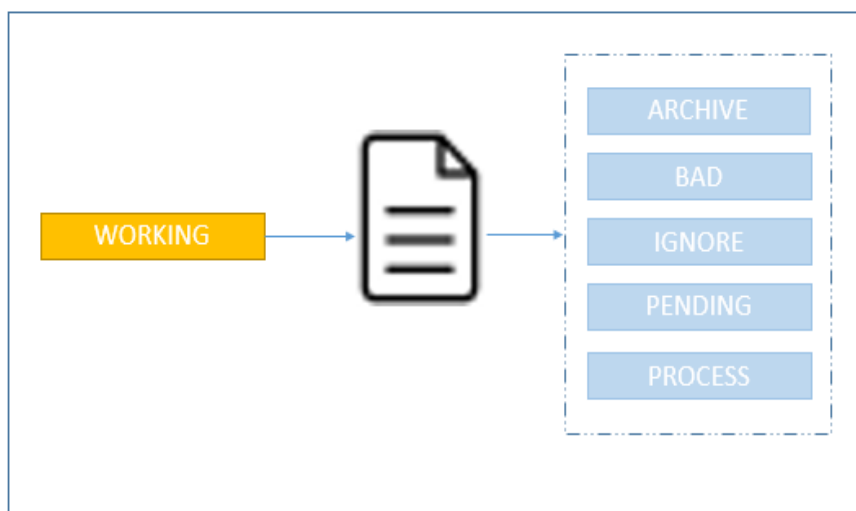   **Max number of files processing parallel is 10.**

2.2.1   Sequential / Serial Processing

- ■ Processing that occurs in sequence and not altogether.

   **In below screen one file is processed at a time.**

2.2.2 *Parallel Processing*

- Data parallelism refers to scenarios in which the same operation is performed concurrently (that is, in parallel) on elements in a source collection or array. In data parallel operations, the source collection is partitioned so that multiple threads can operate on different segments concurrently or in simple words data processing wherein at least two or more chains of functions are carried out at the same time by independent processors.

  MMSLOADER PROCESSES 10 FILES MAX IN PARALLEL

  In below screen Max 10 files are processed concurrently at same time
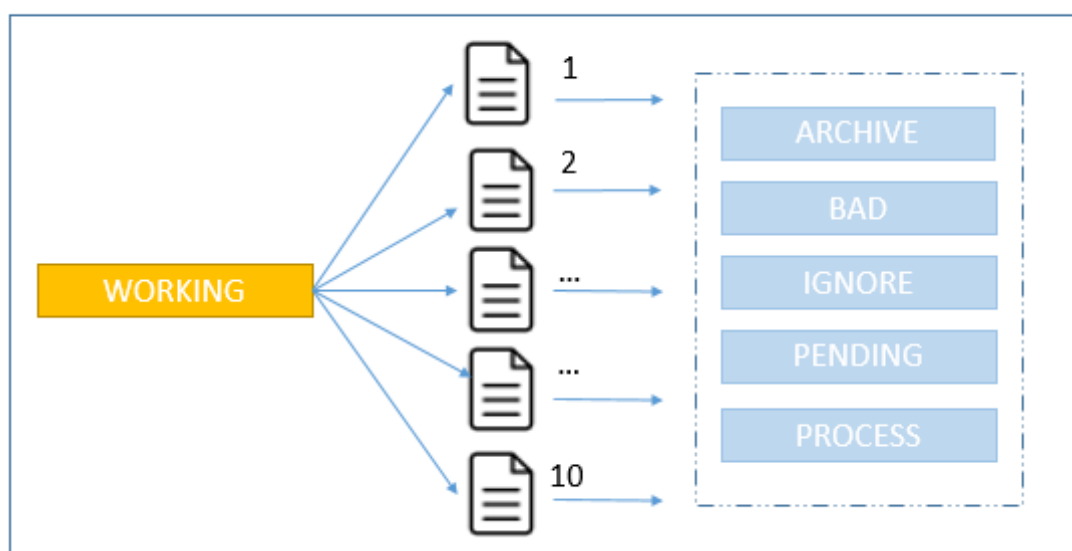
3. LOADER SETUP

Please refer to user guide in separate documet.

4. USER GUIDE

Please refer to user guide in separate documet.

5. CONCLUSION