AIM

Application Functional Design

EETDB Mass Upload Tool

Programme: UNIDO EETDB

Author: Nikolay Komissarenko

Creation Date: 2 August 2013

Last Updated: 20 August 2013

Version: 1.1

1. **Title, Subject, Last Updated Date, Reference Number**, **and** **Version** are marked by a Word Bookmark so that they can be easily reproduced in the header and footer of documents. When you change any of these values, be careful not to accidentally delete the bookmark. **You can make bookmarks visible by selecting Tools->Options…View and checking the Bookmarks option in the Show region.**

**Approvals:**

|  |  |
| --- | --- |
| TBD |  |
| TBD |  |
| TBD |  |

1. To add additional approval lines, press [Tab] from the last cell in the table above.
2. You can delete any elements of this cover page that you do not need for your document. For example, Copy Number is only required if this is a controlled document and you need to track each copy that you distribute.

## Document Control

Change Record

| Date | Author | Version | Change Reference |
| --- | --- | --- | --- |
|  |  |  |  |
| 2 August 2013 | Nikolay Komissarenko | 1.0 | draft |
| 20 August 2013 | Nikolay Komissarenko | 1.1 | Technical details added |

Reviewers

| Name | Position |
| --- | --- |
|  |  |
|  | Approver |
|  | Reviewer |
|  | Approver |

1. The copy numbers referenced above should be written into the **Copy Number** space on the cover of each distributed copy. If the document is not controlled, you can delete this table, the Note To Holders, and the **Copy Number** label from the cover page.

References

| Document Title | Description | Owner | Location |
| --- | --- | --- | --- |
|  |  |  |  |
| EETDB DB specification | Defines EETDB database structure and code |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### Contents

Document Control  
 ii

Contents iv

Overview v

Definitions v

Assumptions v

Basic Needs vi

Current state vi

Common requirements vi

EETDB Data Access Services requirements vi

EETDB Data Access Services architecture vii

EETDB API ix

Data Access Layer x

Configuration xi

Open and Closed Issues for this Deliverable xiii

Open Issues xiii

Closed Issues xiii

Appendix A xiv

1. To update the table of contents, put the cursor anywhere in the table and press [F9]. To change the number of levels displayed, select the menu option Insert‑>Index and Tables, make sure the Table of Contents tab is active, and change the Number of Levels to a new value.

## Overview

### Definitions

EETDB – Energy Efficient Technologies Data Bank

UNIDO – United Nations Industrial Development Organization

DAS – Data Access Services

CRUD – Create, Read, Update and Delete

MVC – Model, View, Controller

REST - Representational State Transfer

### Assumptions

1. EETDB is not a standalone web-enabled system, it’s part of the UNIDO web site <http://energy.unido.ru/>
2. If you use a user-friendly name for this customization as the replacement for <Subject>, the following paragraphs will default nicely.

## Basic Needs

### Current state

At present EETDB Admin Tools allow a system user to manage and maintain EETDB data by creating, deleting or updating the database entities. The Admin Tools UI provides a convenient and simple way to update or create a single entity. However it’s not suitable for managing or creating multiple entities. In some cases it’s required to add a set of new entities to the system, for example, equipment catalogue that can contain hundreds of records.

### Common requirements

The system should provide a convenient and easy to use way to add/update multiple EETDB entities.

### EETDB Mass Upload Tool requirements

The Mass Upload Tool is part of EETDB solution and is aimed to provide an ability to upload data files to the system in the following file formats:

* Comma Separated Values (\*.csv)
* Microsoft Excel (\*.xls, \*.xlsx)

The data files should comply with the EETDB Data Format Specification.

The Mass Upload Tool should be easily extendable to support other file formats if required (for example (XML).

The Mass Upload Tool should support different transport protocols for pushing data files to the system, the minimal set is:

* By sending a data file in email attachment (SMTP)
* By uploading via FTP
* By uploading on the EETDB web pages (HTTP upload)

The Mass Upload Tool should validate the data file to make sure it complies with the Data Format Specification. When parsing the file and adding data to the EETDB database all errors should be logged and then displayed in a report on the web pages. All successfully parsed records should be added to the database.

### Tools and technologies

Development tools:

* Java Eclipse
* MySQL Workbench

Platform and technologies:

* Java 1.6
* Spring Framework
* Hibernate ORM Framework
* RDBMS: MySQL

## EETDB Mass Upload Tool architecture

The Mass Upload Tool is a Java-based multithread service. The main thread is a container for hosting services responsible for handling incoming data files. The Service Host is configurable to run any number of services implementing Service interface. The basic implementation involves running two main services:

* POP3 Daemon which is responsible for monitoring email inbox
* File System Daemon which is monitoring a file system folder for incoming data files.



When receiving a data file the transport layer handlers use Parsers to extract EETDB data from the file and then pass the extracted data to the DB Helper to persist the data.

POP3Daemon is monitoring a configurable email inbox and in case there is an incoming email it tries to save mail attachments to the folder monitored by the File System Daemon which does rest of the work.



File System Daemon is monitoring a system folder (configurable) for incoming data files. If any files appear in the folder the service checks if the files are of the supported file format. Files of the supported formats get parsed by an appropriate parser and then persisted to the DB. Non supported files get moved to the Garbage folder for further manual inspection.



There is no dedicated service for FTP transport. This requirement is fulfilled by configuring FTP server so the incoming folder is the one monitored by the File System Daemon.

## Mass Upload Tool technical specification

#### Services

The main interface

#### 

#### Parsers

The initial implementation of the Mass Upload Tool supports Comma Separated Values (CSV) and Microsoft Excel file formats. However the architecture

#### DB Helper

## Configuration

There are three main configuration files for DAS:

* web.xml: provides basic Spring MVC configuration parameters and normally shouldn't be changed;
* rest-servlet.xml: provides DAS servlet specific configuration parameters. The only parameter that can be the matter of interest for the support personnel is the following:

<bean id=*"multipartResolver"* class=*"org.springframework.web.multipart.commons.CommonsMultipartResolver"*>

<property name=*"maxUploadSize"* value=*"10000000"* />

</bean>

*maxUploadSize* defines the maximum allowed size for uploaded binary data like images, documents etc.

* rest-context.xml: provides custom DAS parameters like DB connection details etc.

Database connection details are configured in this section:

<bean id=*"dataSource"* class=*"org.apache.commons.dbcp.BasicDataSource"*

destroy-method=*"close"*>

<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"* />

<property name=*"url"* value=*"jdbc:mysql://IP/eetdb?characterEncoding=UTF-8&amp;noAccessToProcedureBodies=true"* />

<property name=*"maxActive"* value=*"-1"* />

<property name=*"maxIdle"* value=*"1"* />

<property name=*"username"* value=*""* />

<property name=*"password"* value=*""* />

</bean>

Hibernate specific parameters can be amended in this section:

<bean id=*"sessionFactory"* class=*"org.springframework.orm.hibernate4.LocalSessionFactoryBean"*>

<property name=*"dataSource"* ref=*"dataSource"* />

<property name=*"mappingResources"*>

<list>

<value>Entity.hbm.xml</value>

<value>EntityProperty.hbm.xml</value>

<value>EntityTemplate.hbm.xml</value>

<value>EntityTemplateProperty.hbm.xml</value>

<value>ValueType.hbm.xml</value>

<value>Topic.hbm.xml</value>

<value>EntitySearchResult.hbm.xml</value>

<value>ValueBlob.hbm.xml</value>

</list>

</property>

<property name=*"hibernateProperties"*>

<props>

<prop key=*"hibernate.dialect"*>org.hibernate.dialect.MySQLDialect</prop>

<prop key=*"hibernate.show\_sql"*>true</prop>

<prop key=*"hibernate.format\_sql"*>true</prop>

<prop key=*"hibernate.max\_fetch\_depth"*>1</prop>

<prop key=*"hibernate.cache.region.factory\_class"*>org.hibernate.cache.ehcache.SingletonEhCacheRegionFactory</prop>

<prop key=*"hibernate.cache.use\_query\_cache"*>true</prop>

<prop key=*"hibernate.cache.use\_second\_level\_cache"*>true</prop>

<prop key=*"configurationResourceName"*>ehcache.xml</prop>

<prop key=*"hibernate.connection.charSet"*>utf8</prop>

</props>

</property>

</bean>

#### Logging

Logging in provided via Log4j and configured via log4j.properties file located in WebContent/WEB-INF/classes folder.

## Deployment

DAS is distributed in a web archive (WAR) file. Deployment of DAS is a standard procedure for WARs – the file should be copied in to the Tomcat webapp/ folder and then it will get installed by the server.

After the installation is complete it can be required to update DAS config files (ref:Configuration).

## Open and Closed Issues for this Deliverable

1. **Define Auto Cash Rules:** Dell Prepaid Rule, Dell Standard RuleAdd open issues that you identify while writing or reviewing this document to the open issues section. As you resolve issues, move them to the closed issues section and keep the issue ID the same. Include an explanation of the resolution.  
     
   When this deliverable is complete, any open issues should be transferred to the project- or process-level Risk and Issue Log (PJM.CR.040) and managed using a project level Risk and Issue Form (PJM.CR.040). In addition, the open items should remain in the open issues section of this deliverable, but flagged in the resolution column as being transferred.

### Open Issues

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

### Closed Issues

| ID | Raised by | Issue | Resolution | Owner | Target Date | Impact Date |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| 1 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

## Appendix A

**JSON examples**

**Entity Data Object example**