SDMX Dataloader

Technical Documentation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Type of Document** | Technical documentation | | | | |
| **Reference:** |  | | | | |
| **Issue:** |  | **Revision:** |  | **Status:** | Draft |
| **Created by:** | Valerio Proietti | | | **Date:** | 03/12/2014 |
| **Updated by:** |  | | |  |  |
| **Approved by:** |  | | | | |

**Document Change Record**

|  |  |  |
| --- | --- | --- |
| Issue/ Revision | Date | Change |
| Start | 03/12/2014 | First draft |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table of Contents

[1 Overview 4](#_Toc405908646)

[2 Indice progetti della soluzione 5](#_Toc405908647)

[3 Procedura di Importazione dati 6](#_Toc405908648)

[3.1 Flowchart 7](#_Toc405908649)

[3.2 Implementazione WebGUI 8](#_Toc405908650)

[3.3 Tabella metodi esposti web service inserimento 9](#_Toc405908651)

[3.4 Esempio richiesta http utilizzando AjaxRequestManager 10](#_Toc405908652)

[4 DB Access Layer 11](#_Toc405908653)

[4.1 DataAccess 11](#_Toc405908654)

[4.2 DataSDMX 12](#_Toc405908655)

[5 Analisi funzioni principali 14](#_Toc405908656)

[6 File supportati 16](#_Toc405908657)

[7 Schema DDB 17](#_Toc405908658)

# Overview

Documentazione tecnica per SDMX DataLoader, descrive tecnologie e metodi implementati.

|  |  |
| --- | --- |
| Acronimo | Termine |
| DDB | **Dissemination Database**, contiene i dati Delle serie storiche importate. |
| MR | **Metadata Repository**, contiene metadati strutturali SDMX |
| WebGUI | Interfaccia di inserimento composta da controlli di vario tipo. Può essere implementata in vari contesti dalle WebGUI, ai Windows Form |

# Indice progetti della soluzione

|  |  |
| --- | --- |
| Progetto | Classi |
| ISTAT.DBDAL | DataWrapper  DataAccess  DataSDMX |
| ISTAT.ENTITY | Attribute  Category  Code  Codelist  Concept  ConceptDB  DataStructure  Dimension  QueryPart  TextTypeWrapper  Transcode  UserDef |
| ISTAT.EXPORT | Exporter  SettingsHandler |
| ISTAT.IO | CSVReader  CSVWriter  Utility |
| SDMX\_Dataloader.Engine | Mapping  ProcManager  Client |

Tabella Indice classi

# Procedura di Importazione dati

Gestita con tecnica Ajax l’intera procedura non necessita di post back.

Nello schema è rappresentata l’iterazione tra client e server.

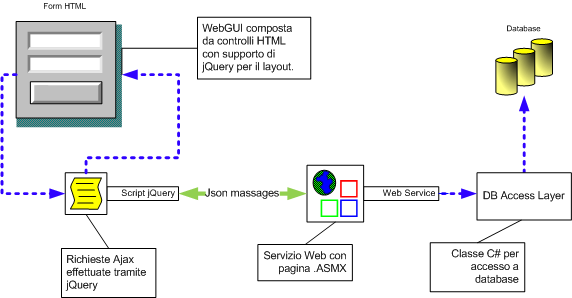


Figura - Iterazione client server

## Flowchart

Di seguito è rappresentato tramite schema a blocchi la procedura per l’importazione dati.

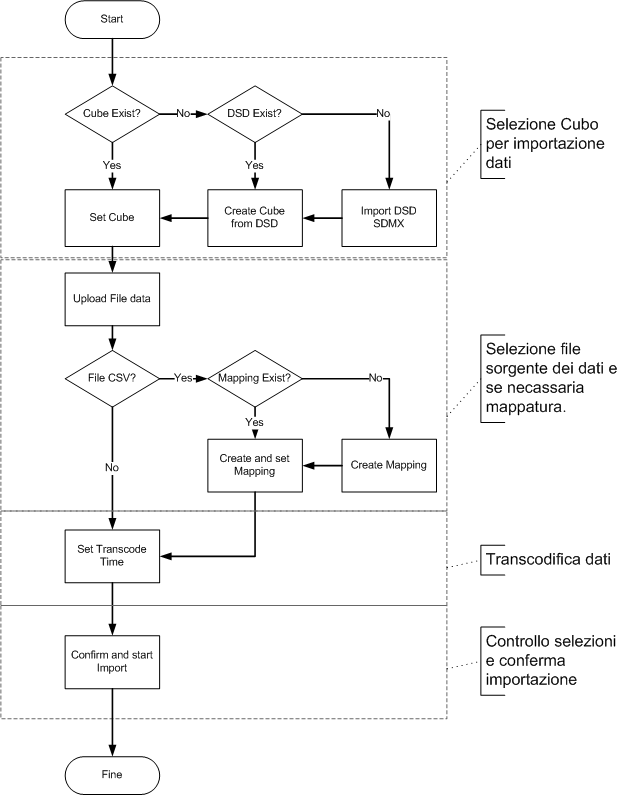


Figura - Flowchart procedura importazione

## Implementazione WebGUI

La WebGUI è stata implementata usando il plugin jQuery-UI la cui documentazione è disponibile all’indirizzo http://jqueryui.com /.

## 

## Tabella metodi esposti web service inserimento

Questa tabella mostra i metodi e i relativi parametri richiamabili tramite richieste http POST.

|  |  |
| --- | --- |
| Metodo | Utilizzo |
| GetDSD(string agency, string id, string version) | Metodo per inizializzare oggetti |
| GetCodelist(string code, bool isDimensionCode = true) | Metodo per ottenere tutte le info per una DSD |
| SetDataset(int idset) | Metodo per ottenere tutte le info per una DSD |
| CreateDataSet(string code,int idtheme, List<string[]> names, List<string> dimensions, List<string> attributes) | Metodo per settare la DSD selezionata |
| DeleteDataset(int idset) | Metodo per creare la dataset |
| InitMapping() | Metodo per cancellare un DS |
| GetMappings() | Metodo per inizializzare oggetti |
| GetDimAtt() | Metodo per ottenere lista dei mapping |
| GetMappingSchema(int idschema) | Metodo per ottenere dimensioni e attributi da un cubo |
| CreateMapping(string name, List<SDMX\_Dataloader.Engine.MappingItem> items) | Metodo per ottenere mapping esistente |
| DeleteMapping(int idschema) | Metodo per creare mapping |
| GetTranscodeTime(bool useTranscode, int period, string espCurr, char charPeriod) | Metodo per cancellare Mapping |
| CreateExport(string agency, string id, string version, List<string> dimensions, List<string> attributes, string type\_download) | Metodo per impostare il tipo di conversione del campo TIME\_PERIOD |
| UploadFile() | Metodo che permette di creare file zip contenente export per .STAT |
| GetCurrentHeaders() | Metodo per l’upload di file CSV |
| UploadFileSDMXSchema() | Metodo per ottenere gli header del file CSV |
| UploadFileSDMXData() | Metodo per l’upload e l’import di file di struttura SDMX |
| GetPreview() | Metodo per l’upload di file dati SDMX |
| ImportData(string typeImport,string emailToReport) | Metodo per ottenere preview dei dati caricati |
|  | Metodo per effettuare l’import finale dei dati |
|  |  |

Tabella 2 - Metodo esposti in WebServices/WImport.asmx

## Esempio richiesta http utilizzando AjaxRequestManager

AjaxRequestManager (./scripts/Utility.js) è un oggetto javascript che si occupa di settare e rendere più flessibile l’iterazione con il web services.

<script type="text/javascript">

var args={

arg1=0,

arg2=1

}

// Init object for request

var \_ajaxRequestMan=new AjaxRequestManager(

{

baseUrl:”http:\\localhost\webservice.asmx”,

method:”webMethod”,

locale:"en"

},{

titleLoading: "Please wait",

messageLoading: "Wait end operation.",

});

// Send request

\_ajaxRequestMan.SendRequest(

args,

function(jsonResult){

// Success calback

\_layoutManager.DisableLoadingDialog();

},

function(msg){

// Error callback

\_layoutManager.DisableLoadingDialog();

},

function(msg){

// Start callback

\_layoutManager.EnableLoadingDialog();

\_layoutManager.InitControl();

},

function(msg){

// End callback

\_layoutManager.DisableLoadingDialog();

},

true);

</script>

L’azione \_ajaxRequestMan.SendRequest può essere associata a qualsiasi evento che sia la pressione di un bottone o il cambiamento di un item in una listview.

Ad ogni oggetto AjaxRequestManager è associato un webMethod che se riceve i giusti parametri risponderà con una stringa json automaticamente convertità in un oggetto javascript accessibile dal parametro jsonResult del callBack di successo operazione.

# DB Access Layer

Il progetto ISTAT.DBDAL contiene le classi che implementano e regolano l’accesso al database.

La classe DataWrapper contiene la proprietà

public System.Data.IDbConnection DBConnection { get { return IConn; } }

IConn e’ un interfaccia di tipo System.Data.IDbConnection che viene istanziata tramite il costruttore della classe da qui ottiene tipo e stringa di connessione.

**NB: Al momento supporta connessioni SQL Server versione 2008 o superiori.**

## DataAccess

La classe DataAccess effettua tramite funzioni statiche richiami a stored procedure su database “Datawarehouse”, che rappresenta il db di diffusione dati.

Prendimo in esame la funzione Get\_Themes che restituisce la lista di categorie disponibili

public static List<Category> Get\_Themes() {

DataWrapper dtw =

new DataWrapper(DataWrapper.ECONNECTIONTYPE.SQL,

DataAccess.SQLConnString\_DB.ConnectionString);

if (dtw.TestConnection())

{

dtw.DBConnection.Open();

try

{

System.Data.IDbCommand cmd = dtw.DBConnection.CreateCommand();

cmd.CommandType = System.Data.CommandType.StoredProcedure;

cmd.CommandText = "dbo.proc\_GET\_THEME";

List<Category> themes = new List<Category>();

IDataReader \_reader=cmd.ExecuteReader();

while(\_reader.Read()){

Category theme = new Category(

\_reader.GetInt32(\_reader.GetOrdinal("IDTheme")),

\_reader.GetInt32(\_reader.GetOrdinal("IDThemeParent")));

List<TextTypeWrapper> names= DataAccess.Get\_LocalisedStrings(.IDTheme,CatTheme");

theme.Names.Add(

names[TextTypeWrapper.GetIndexLocale(names, DataAccess.TwoLetterIso)]);

themes.Add(theme);

}

dtw.DBConnection.Close();

return themes;

}

catch

{

dtw.DBConnection.Close();

return null;

}

} return null;

}

Corrispettivo pseudo codice della funzione Get\_Themes

Creo nuova connessione verso DDB

Se (connessione supera il test)

Apro connessione verso DDB

Provo

Creo un comando da eseguire su DDB

Dico al comando che deve eseguire una procedura registrata già sul server

Aggiungo parametri richiesti al comando

Creo contenitore risultati

Finche(Il comando ottiene dei risulti)

Ottengo tutti i dati necessari

Aggiungo risultati nel contenitore

Chiudo connessione

Restituisco i risultati ottenuti finora.

Se genero eccezione

Chiudo connessione

Restituisco risultati nulli

Se non supero test

Restituisco risultati nulli

La maggior parte delle funzioni e’ strutturata in questo modo, da notare che la funzione i caso di fallimento se deve restituire un oggetto ritornerà un valore *Null* se deve restituire un numero il valore nullo sarà *-1* per le stringhe di testo sarà *string.Empty.*

## DataSDMX

La classe DataSDMX ottiene dati da database MR tramite l’utilizzo di SDMX Common API e Richiami a stored procedure sul db.

E’ mostrata un tipico utilizzo dell’oggetto ISdmxMutableObjectRetrievalManager con il quale è possibile ottenere oggetti SDMX.

public static ICodelistObject GetCodelist(SDMXIdentifier sdmxKey)

{

ISdmxMutableObjectRetrievalManager manager = GetManager();

IMaintainableRefObject query =

new MaintainableRefObjectImpl(sdmxKey.agencyid, sdmxKey.id, sdmxKey.version);

ICodelistMutableObject codelist = manager.GetMutableCodelist(query, false, false);

return codelist.ImmutableInstance;

}

Ottenuto il manager si istanzia una query con id,agencyid,e version dell’artefatto richiesto e si richiede l’oggetto Mutable per poi tornare l’istanza Immutable.

Per ottenere l’oggetto manager lo si instanzia tramite il metodo GetRetrievalManager contenuto nella clase MutableRetrievalManagerFactory passandogli la stringa di connessione verso l’MR.

private static ISdmxMutableObjectRetrievalManager GetManager()

{

MutableRetrievalManagerFactory factory =

new Estat.Sri.MappingStoreRetrieval.Factory.MutableRetrievalManagerFactory();

ISdmxMutableObjectRetrievalManager manager = factory.GetRetrievalManager(System.Configuration.ConfigurationManager.ConnectionStrings["MappingStoreServer"]);

return manager;

}

# Analisi funzioni principali

Nella fase finale della procedura viene richiamato il webMethod ImportData (./WebServices/WImport.asmx)

public string ImportData(string typeImport,string emailToReport)

all’interno di essa viene eseguita l’istruzione

Dictionary<string, List<string>> \_view = user.ProcedureManager.Get\_DATA\_VIEW(0);

Attraverso l’oggetto user.ProcedureManager (gestore degli oggetti utili per il compimento della procedura di importazione) si ottiene una collezione di liste, la quale rappresenta la vista da inserire nel DDB.

**NB:** La \_*view* dichiarata come Dictionary<string,List<string>> sarà nella forma normale se ha il CODE del concept cosi formata.

\_view={ “FREQ”=>{“M”, “M”, “M”, “M”, “M”, “M”},

“ADJASTMENT”=>{“1”, “1”, “1”, “0”, “0”, “0”},

“REF\_AREA”=>{“1”, “1”, “2”, “2”, “3”, “3”},

“TIME\_PERIOD”=>{“1990M01”, “1990M02”, “1990M01”, “1990M02”, “1990M01”, “1990M02”},

“OBS\_VALUE” =>{“1.2”, “.2”, “2.3”, “4.1”, “5.7”, “6”},

“OBS\_STATUS” =>{“1”, “2”, “3”, “4”, “5”, “6”}}

Ottenuta la view da inserire segue la seguente riga di codice

QueryReport \_result=DataAccess.Insert\_Data(user.ProcedureManager.DataSet.IDSet, \_view);

Tramite DataAccess richiamiamo il metodo Insert\_Data a cui passiamo i parametri id del dataset su cui si vuole effettuare l’inserimento e la vista generata.

Verrà restituito l’oggetto QueryReport che contiene info sull’esito dell’operazione.

public static QueryReport Insert\_Data(int idset,Dictionary<string,List<string>> view)

Di seguito lo pseudo codice implemento nella funzione.

Per ogni riga nella VIEW

Per ogni dimensione nella lista delle dimensioni

Se la dimensione non è una TIME\_DIMENSION

Se non ho già risolto l’id del CODE della dimensione

IDDIM = ottengo l’id del CODE

se l’ IDDIM e’ stato trovato

Aggiorno l’attuale FILT\_QUERY\_PART dal database

Altrimenti registro errore

Altrimenti se ho già risolto questo CODE

Aggiorno l’attuale FILT\_QUERY\_PART dalla precedente

Se non ho errori

Per ogni attributo nella lista delle attributi

Se non ho già risolto l’id del CODE del attributo

Se l’attributo è una CODELIST

IDATT=Ottengo l’id del CODE

Altrimenti

Inserisco il nuovo CODE

IDATT=Ottengo l’id del CODE

Se l’attributo e’ un OBSERVATION\_VALUE

Aggiorno l’attuale FACT\_QUERY\_PART con l'IDATT

Altrimenti

Aggiorno l’attuale FILT\_QUERY\_ PART con l’IDATT

Altrimenti se ho già risolto questo CODE

Aggiorno l’attuale FACT \_QUERY\_PART dalla precedente

Se non ho errori

Se l’attuale FILT\_QUERY\_PART è diversa dalla precedente

SID=ottengo l’SID dal DDB tramite la combinazione di FILT\_QUERY\_PART

Aggiorno la precedente FILT\_QUERY\_ PART con l’attuale

Se ho un SID valido

IDTIME=ottengo l’id del TIME\_PERIOD

Se ho un IDTIME valido

OBS\_VALUE=value nella VIEW

Inserisco l’IDTIME, l’OBS\_VALUE e FACT\_QUERY\_PART del DDB

Altrimenti registro errore

Altrimenti registro fallimento inserimento riga i-esima

# File supportati

SDMX Dataloader supporta l’upload di file dati del seguente tipo:

File .CSV: File di testo UTF8 con carattere di separazione per i campi e carattere di “endline” per fine riga.

*REFERENCE\_AREA;SUBJECT;MEASURE;ADJUSTMENT;FREQ;UNIT;TIME\_PERIOD;OBS\_VALUE*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-01;101.2*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-10;103.8*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-11;103.7*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-12;104.1*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-02;101.5*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-03;101.9*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-04;102.4*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-05;102.5*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-06;102.6*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-07;102.9*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-08;103.2*

*IT;CPALTT01;IXNB;NSA;M;2010100;2011-09;103.2*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-01;104.4*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-10;106.5*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-11;106.3*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-12;106.5*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-02;104.8*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-03;105.3*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-04;105.8*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-05;105.8*

*IT;CPALTT01;IXNB;NSA;M;2010100;2012-06;106*

File .XML: File dati SDMX in formato 2.0 Compact, 2.0 Generic, 2.1 Structure Specific,2.1 Generic

SDMX Dataloader supporta l’upload di file di struttura del seguente tipo:

File .XML: File SDMX Structure

# Schema DDB

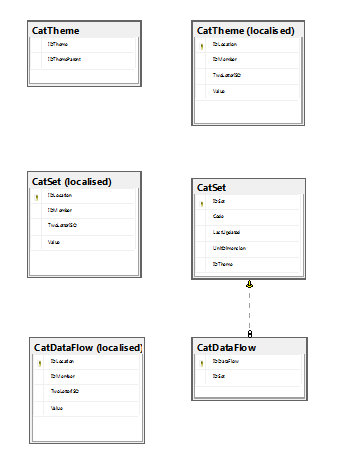


Figura - Schema DDB dataset

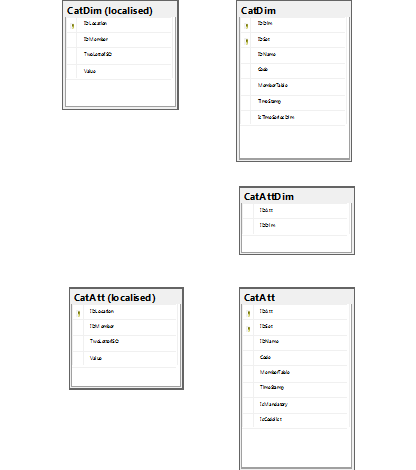


Figura 4 - Schema DDB Dimension/Attribute

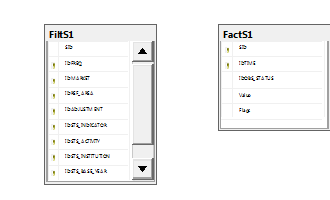


Figura - Schema DDB Cube

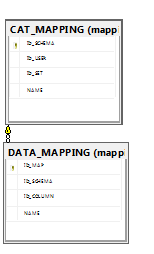


Figura - Schema DDB mapping

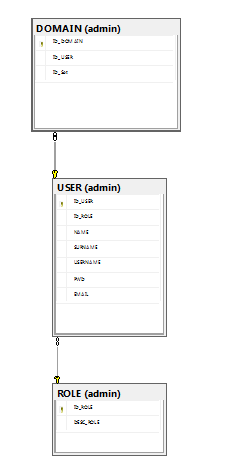


Figura - Schema DDB users