



SDMX self-learning package No. 7

Student book

## **SDMX Architecture Using the Pull Method for Data Sharing**

Produced by	Eurostat, Directorate B: Statistical Methodologies and Tools Unit B-5: Statistical Information Technologies
Last update of content	September 2010
Version	1.0

---

# TABLE OF CONTENTS

<b>1</b>	<b>SCOPE OF THE STUDENT BOOK.....</b>	<b>1</b>
<b>2</b>	<b>THE DATA SHARING MODEL .....</b>	<b>2</b>
2.1	ACTORS .....	2
2.2	SCHEMES FOR DATA TRANSMISSION USING THE PULL MECHANISM .....	2
<b>3</b>	<b>ARCHITECTURES .....</b>	<b>4</b>
3.1	DATA REPOSITORY (WAREHOUSING) ARCHITECTURE .....	4
3.2	DATA HUB ARCHITECTURE.....	5
3.3	SDMX NSI REFERENCE SERVICE INFRASTRUCTURE .....	6
<b>4</b>	<b>THE MAPPING PROCESS.....</b>	<b>8</b>
4.1	PRE-REQUISITES .....	8
4.2	HARMONISATION OF STATISTICAL CONTENT / MAPPING.....	9
4.2.1	<i>When to carry out mapping and why.....</i>	<i>10</i>
4.2.2	<i>Building an SQL Query .....</i>	<i>10</i>
4.2.3	<i>Creating an SDMX data file .....</i>	<i>12</i>
4.3	THE MAPPING PROCESS – EXAMPLE: CENSUS HUB .....	13
4.3.1	<i>Step 1: List of local concepts.....</i>	<i>14</i>
4.3.2	<i>Step 2: Associating all local concepts with DSD concepts.....</i>	<i>15</i>
4.3.3	<i>Step 3: Mapping codes .....</i>	<i>18</i>
4.3.4	<i>Translation of an SDMX query.....</i>	<i>19</i>
4.3.5	<i>Translation to a dataset.....</i>	<i>20</i>
<b>5</b>	<b>GLOSSARY.....</b>	<b>21</b>

## **1 Scope of the student book**

Both student books No 7 and No 8 deal with the architectures for data sharing as defined by the SDMX (Statistical Data and Metadata eXchange) standard.

In the following chapters, two SDMX architectures will be briefly described, with Eurostat acting as a ‘data collector’ or ‘data consumer’, and ISTAT (National Statistical Institute of Italy), acting as ‘data provider’ or ‘data reporter’.

However, the main focus will be on the harmonisation process — hereafter referred to as the ‘mapping process’ — of the data and metadata that are exchanged between the ‘data provider’ and the ‘data collector’.

<b>Ref.</b>	<b>Title</b>
[01]	Introduction to SDMX
[02]	The SDMX Information Model
[03]	SDMX-ML Messages
[04]	Data Structure Definition
[05]	Metadata Structure Definitions
[06]	XML based technologies used in SDMX
[07]	SDMX architecture using the pull method for data sharing – Part 1
[08]	SDMX architecture using the pull method for data sharing – Part 2

**Table 1 - SDMX Student Books**

### **Prerequisites**

Reading the previous student books is strongly recommended.

## **2 The data sharing model**

The Data Sharing Model is a mechanism by which data (e.g. statistics) are made available to users in a common environment (the Internet) in a common technical format and with agreed common codes and metadata<sup>1</sup>. In this model, a group of organisations agrees on a common way to provide access to data according to standard processes, formats and technologies.

There are two ways of sharing statistical data: the ‘push’ method, in which the data provider takes action to send data to the data collector; and the ‘pull’ method<sup>2</sup>, whereby data are fetched by the data collector directly from the data provider.

The pull method has the advantage of reducing the reporting burden for European and international organisations; in fact a data-reporting organisation publishes data once, and lets their counterparties ‘pull’ data and related metadata as required.

### **2.1 Actors**

There are two actors in the pull method: the data provider (the organisation providing the data) and the data collectors (the organisation using the data made available by the data provider). The process will be described from both sides.

### **2.2 Schemes for data transmission using the *PULL* mechanism**

There are two possible schemes for data transmission using the *PULL* mechanism:

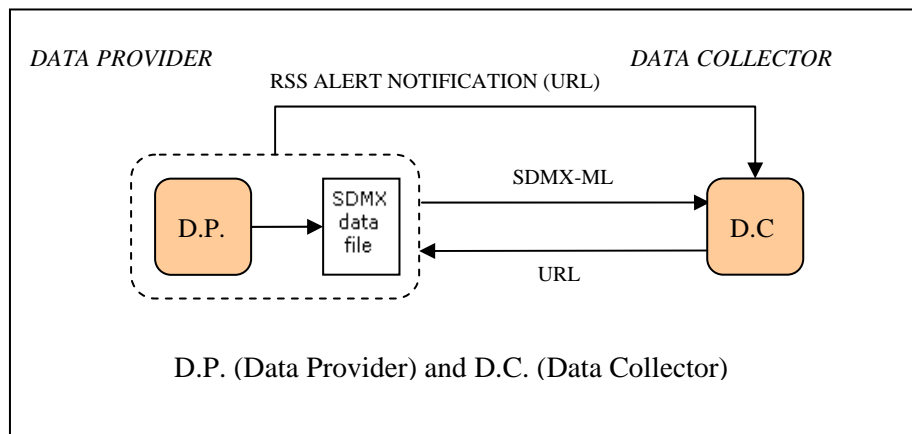
Scenario A (Figure 1)

Data providers prepare newly available data in the form of SDMX files, which are maintained on a specific URL (such as an HTTP server). In this case, there is no need for an SDMX query as in scenario B, because the data reside in a URL that is accessed by Pull Requestor. This module determines whether the feeds contain available data for retrieval. When the new data are prepared, a notification is formulated automatically (in the form of a web feed) and contains a description of the new or updated data. This notification also contains a set of metadata included in the header of the SDMX file, so that the Pull Requestor can identify to which dataflow/metadataflow the dataset belongs to, who is the data provider, as well as the time point or period covered by the data.

---

<sup>1</sup> SDMX uses the following definition of Data Sharing. “Data sharing” is defined as a type of data exchange where open, freely available data formats and process patterns are known and standard. Thus, any organisation or individual can use any counterparty’s data and metadata (assuming they are permitted access to it). This model requires no bilateral agreement, but only requires that data and metadata providers and consumers adhere to the standards.

<sup>2</sup> Reference source: Student book – “Introduction to SDMX” Chapter 3.4.2.3 “Push and Pull Mode”.

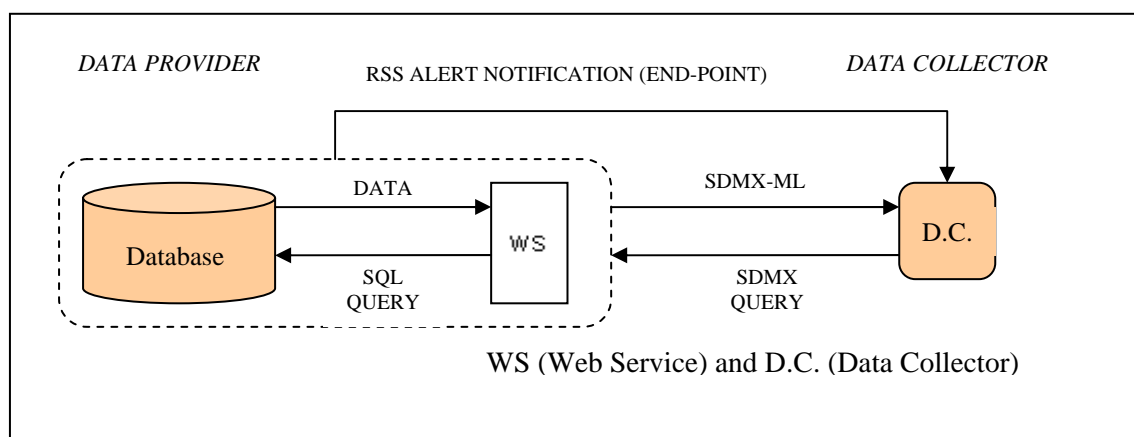


**Figure 1 - Data sharing using SDMX data files**

Scenario B. (Figure 2)

Data providers publish data on their local dissemination data warehouse and provide a web service that can get an SDMX query and respond with an SDMX dataset. This scenario is described in detail in the following paragraphs:

- The **PULL** transmission process starts when the 'Pull Requestor' module periodically checks the web feeds generated by the NSI system. These feeds will reside in the NSI's web servers. If new data are available, the requestor sends an **SDMX-ML** request to the NSI web service, which in turn responds with an **SDMX-ML** dataset.
- Next, the **SDMX-ML** dataset is passed to the **SDMX Web Service** module, where the **SDMX-ML** dataset is validated. The validation includes syntax verification against the corresponding **XML Schemas** and code list validation to be performed directly from these **XML Schemas**.



**Figure 2 - Data sharing using a web service**

### 3 Architectures

In order to facilitate the introduction of SDMX, Eurostat has developed two service infrastructures for data collection that are used in several projects jointly with the Member States:

- a) Data Repository (warehousing) architecture;
- b) Data Hub architecture.

#### 3.1 Data Repository (warehousing) architecture

The Data Repository architecture is implemented by those collecting organisations that periodically collect the data and load them in their databases. In general, a batch process is used to automate the flow in which a whole or partial dataset, including incremental updating, is used.

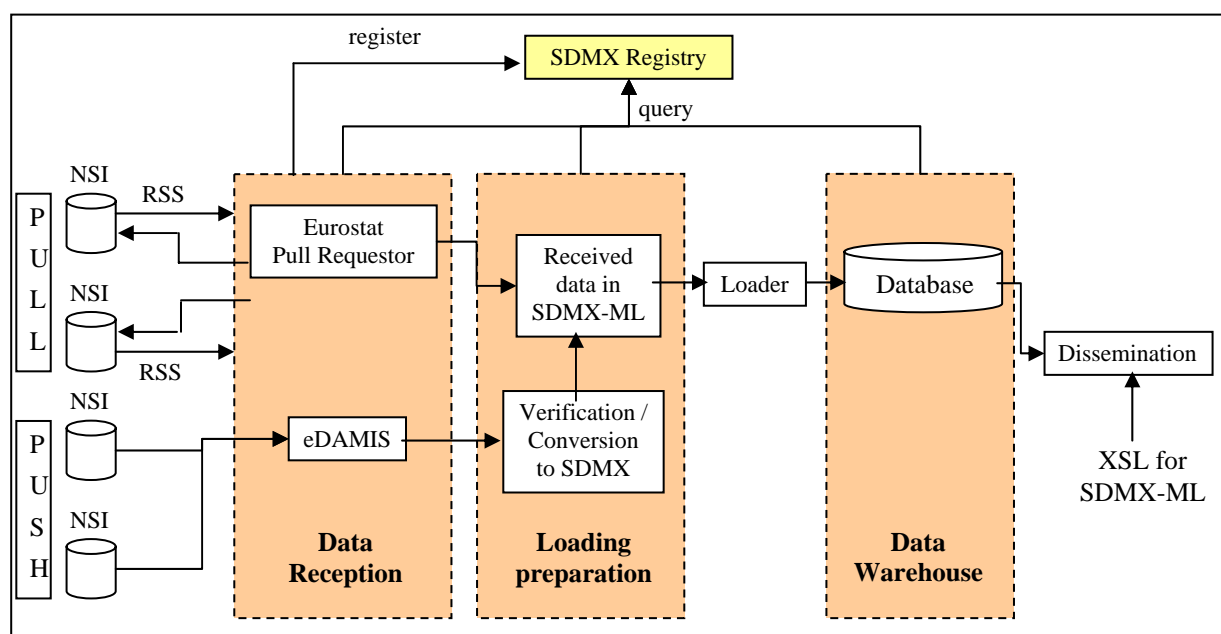
The Data Repository architecture in Figure 3 supports both the ‘push’ and ‘pull’ methods.

The push method within Data Repository architecture expects the data provider (NSI production system) to push data through the Single Entry Point (eDAMIS).

In the context of the pull method, the NSI production system component offers the RSS or Atom feed interface where the data provided by the NSI are listed and also the HTTP/SOAP interface where the data described in the feed can be retrieved either from a URL (HTTP) or SOAP by calling a Web Service that accepts SDMX-ML queries.

The Pull Requestor component checks the RSS/Atom feeds and retrieves data from the NSIs in SDMX-ML format.

The data is forwarded to the SDMX Web Service for dispatching.



**Figure 3 - Data Repository (warehousing) architecture**

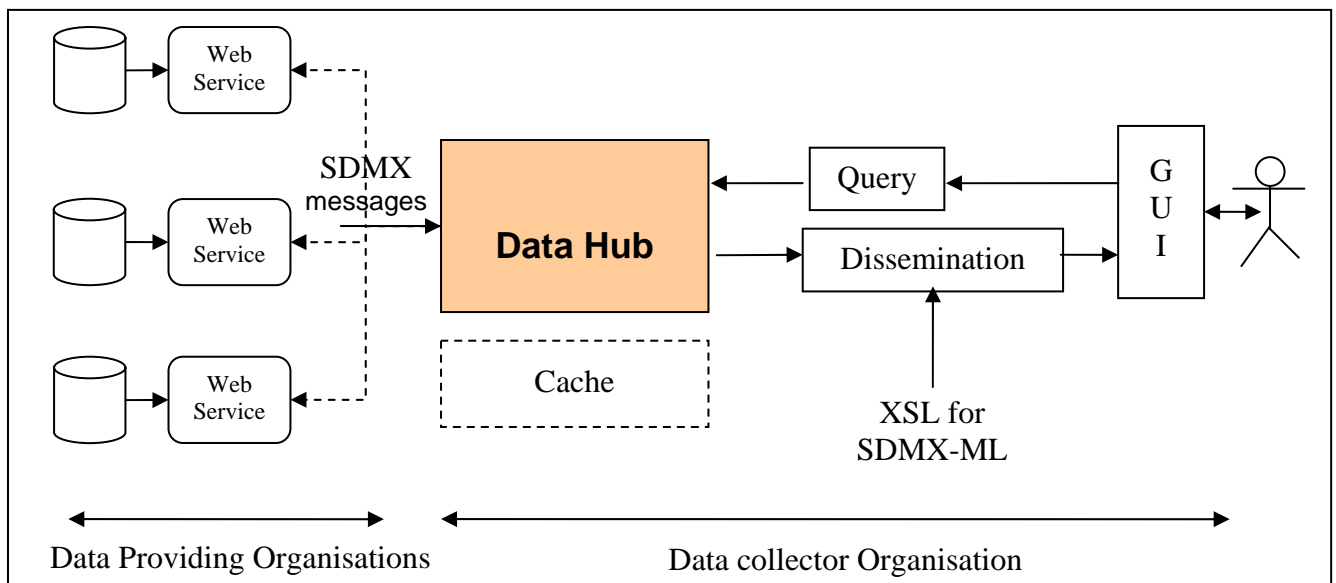
### 3.2 Data Hub architecture

The Data Hub architecture in Figure 4 supports only the ‘pull’ method. This means that a group of partners agree on providing access to their data directly from their database according to standard processes, formats and technologies (web service).

From the data management point of view, the hub is also based on pre-specified datasets, which are — contrary to the database-driven architecture — not kept locally in the central hub system. Instead, the process operates as follows:

- A user identifies a dataset through the Web Graphical User Interface of the Data hub using the structural metadata, and requests it.
- The Data Hub translates the user request in one or more queries and sends them to the related data providers’ systems.
- Data Providers’ systems process the query and send the result to the Data Hub in standard format (SDMX-ML).
- The Data Hub puts together all the results originated in all implicated Data Providers’ systems and presents them in a human-readable format.

Figure 4 below illustrates the Hub architecture:

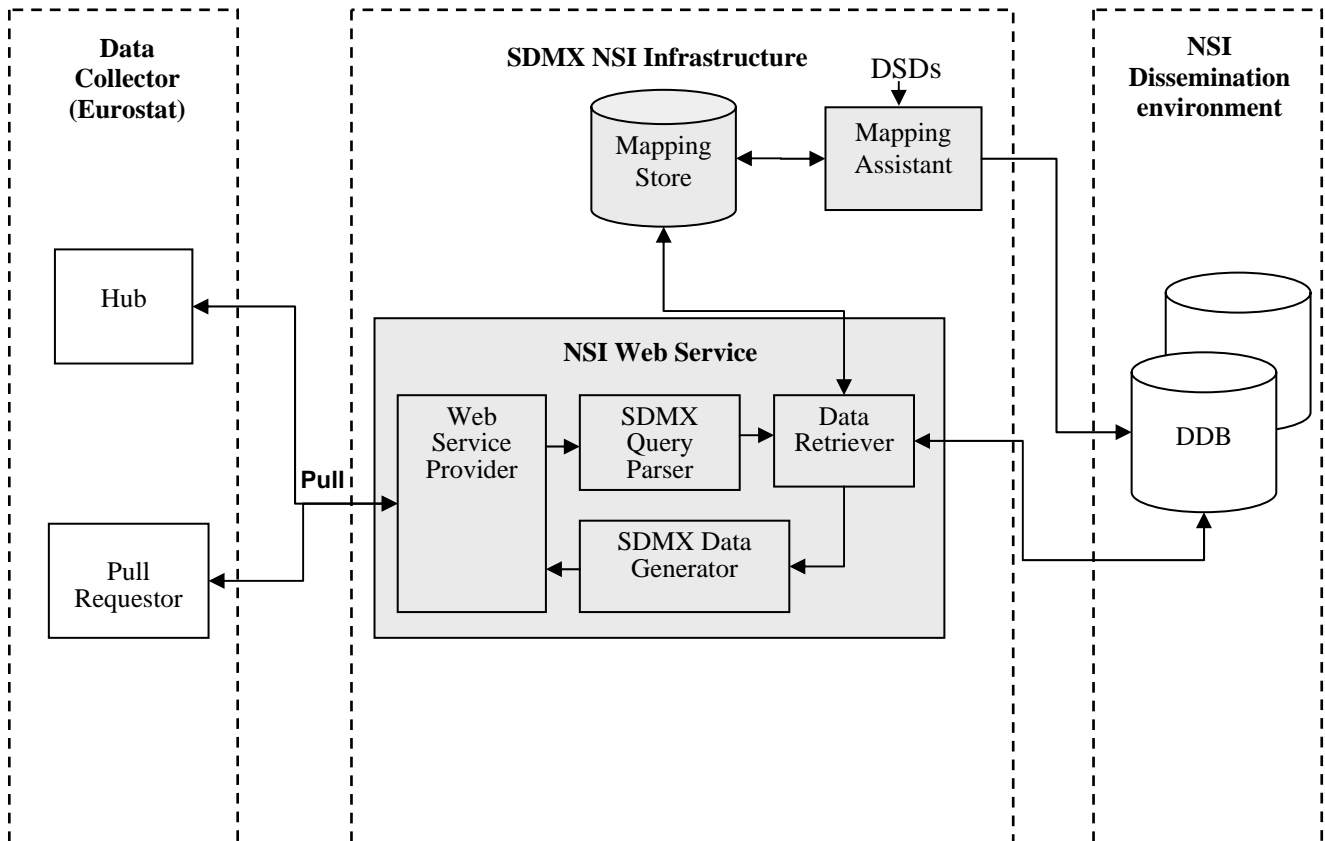


**Figure 4 - Data Hub architecture**



### 3.3 SDMX NSI Reference service infrastructure

The architecture described below in Figure 5 uses the open source architecture developed by Eurostat (SDMX NSI Reference service infrastructure).



**Figure 5 - Simplified view of the SDMX NSI Reference service infrastructure**

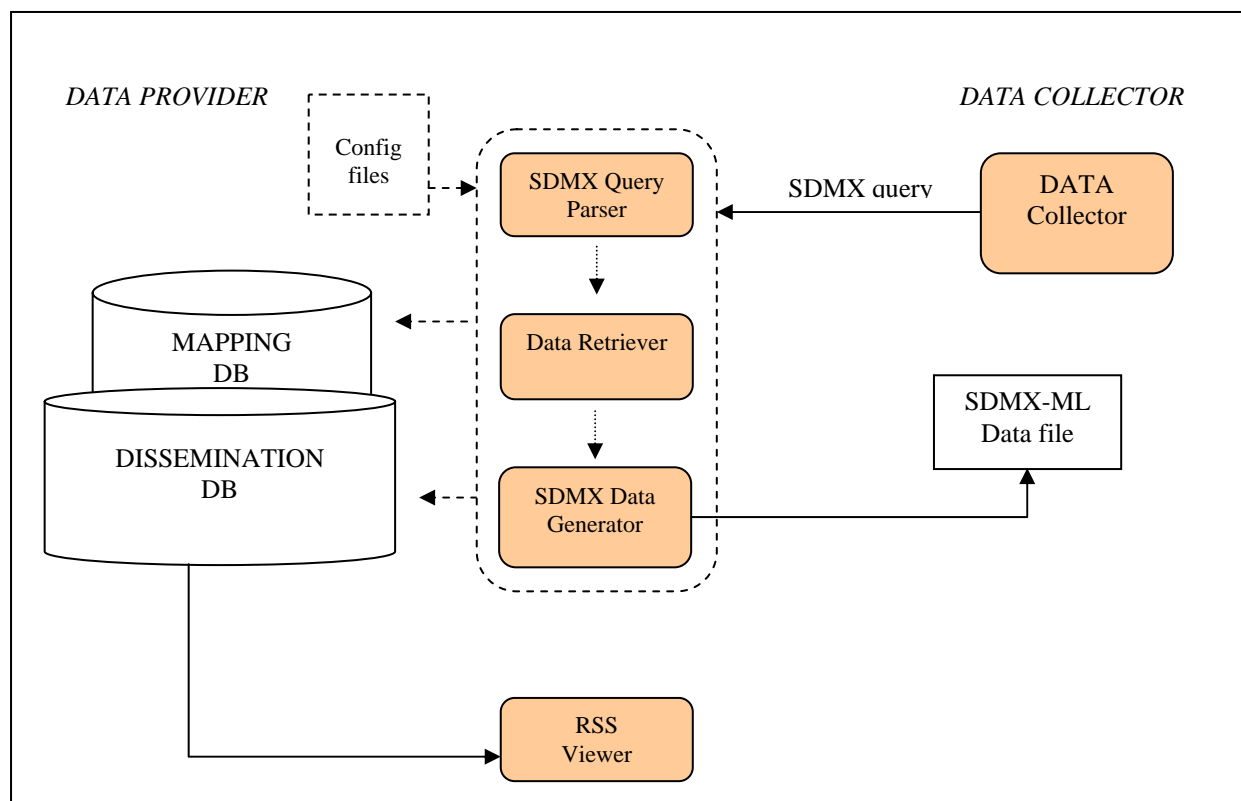
Within this environment, the data provider stores the data in two different dissemination database architectures:

- One database with the dissemination data. In this database, it is recommended to create the structural metadata using the SDMX DSD as a reference. The tables contain concepts, code lists and all the information contained in the DSD and the data files.
- Another database structured to store the mapping rules between local concepts of the dissemination DB and the concepts of the DSD. In this database it is necessary to load the DSD with all the mappings between local structural metadata and SDMX structural metadata.

Providing a web service interface, the architecture is made up of three APIs:

- The ‘SDMX Query parser’, which receives the SDMX query as input and translates it to an internal SDMX query model.
- The ‘Data Retriever’, which performs the SQL query for the local database using the mapping information to SDMX data structures. (This is the only module that accesses the databases).
- The ‘SDMX Data Generator’, which builds the SDMX-ML data files.

The Schema of web services is presented in Figure 6:



**Figure 6 - Schema of web services**

## **4 The Mapping process**

### **4.1 *Pre-requisites***

In order to implement the SDMX IT architecture for data-sharing using the pull mode, several steps must be accomplished by the data provider and a number of questions must be considered:

- Which statistical domains are involved and where are the data currently stored?
- Which structural metadata are involved, and where are they currently stored?
- What is the business process behind the data flow involved?
- Will the SDMX data provider architecture be part of the data warehouse architecture, of the data hub architecture or of both?

Generally, the data and structural metadata involved in the new SDMX information system will either be stored in databases or in files. These two cases lead to different architectural approaches:

- a) Data and structural metadata continue to be stored in files (for example: XLS, CSV, etc.) and the only need is to translate those files into SDMX-ML data files to be pulled by the data collector.
- b) Data and structural metadata are already stored in a database and it is necessary to build suitable software interfaces in order to make the system 'SDMX-compliant'.
- c) A separate special-purpose database is set up to store data and structural metadata. This database will be designed with the main aim of forming part of an SDMX-compliant system. In this case the database can be modelled using the SDMX Information Model.

Cases (b) and (c) make it possible:

- to extract SDMX-ML files from the database that are made available to be pulled by data collectors;
- to query the database directly through a web service.

As explained below, whichever type of data provider architecture is involved, a mapping process between structural metadata may become necessary.

## 4.2 Harmonisation of statistical content / Mapping

Data are generally described differently by data providers and by data collectors, who may use different concepts and code lists.

One of the main purposes of SDMX is the harmonisation of structural metadata. When a new SDMX project begins, one first needs to define the necessary structural metadata (DSD) that will describe data. Generally, this task is performed by the organisation leading the project. Common structural metadata make it possible to exchange data among all actors in a way that can be understood by all. Unfortunately, a typical situation is that data providers' data is already described by local<sup>3</sup> metadata in their databases. The first step is therefore to map the local structural metadata present in the data provider's system against those provided by the DSD.

Table 2 explains the notation used in the coming chapters to describe the mapping process:

Term	Description	Notation
Dimensions, attributes	The dimensions of DSD. Make the assumption that they are coded.	$\mathbf{X}_{\text{dsd}}$ NOTE: X may be any letter except T and M which are reserved (see in this table).
Time dimension	The time dimension of DSD.	$\mathbf{T}_{\text{dsd}}$
Primary measure	The primary measure of the DSD.	$\mathbf{M}_{\text{dsd}}$
Code	The codes used by the DSD for its dataflow.	$\mathbf{x1}_{\text{dsd}}, \mathbf{x2}_{\text{dsd}}, \dots$ NOTE: $\mathbf{x}_{i\text{dsd}}$ is the $i^{\text{th}}$ code of component x.
Local concepts	The local concepts.	$\mathbf{X}_{\text{loc}}$
Local time dimension	The local concept that contains the time of the data.	$\mathbf{T}_{\text{loc}}$
Local code	The codes that are used in the local concepts. Make the assumption that it is not coded.	$\mathbf{x}_{\text{loc}}$ NOTE: $\mathbf{x}_{i\text{loc}}$ is the $i^{\text{th}}$ code of local component x.
Primary key	The primary key of the table containing data. Can be made up of one or more of the table's columns.	<b>PK</b>
Foreign key	The foreign key of a database table containing other local concepts associated to the data. Can consist of one or more of the table's columns.	<b>FK</b>
Column Name	Column name of the $i^{\text{th}}$ column containing the observed value.	$\mathbf{CN}_i$
Table Name	Table name of $i^{\text{th}}$ table containing the observed value.	$\mathbf{TN}_i$

**Table 2 - Notation for the mapping process**

<sup>3</sup> The term “local” is used to indicate that the structure metadata are not SDMX-compliant. In general, they are only valid for the system in which they are stored.

### 4.2.1 When to carry out mapping and why

A mapping process becomes necessary when the data provider must disseminate SDMX format data that are described by local<sup>4</sup> structural metadata. To illustrate the mapping process, the following scenario has been used:

- The data providers' data are described by local concepts that are different<sup>5</sup> from the concepts of the corresponding DSD.
- The local concepts and DSD concepts are in a one-to-one relationship. In order to simplify the examples, the same letters are used for the same concepts on both sides, e.g. local concept  $C_{loc}$  corresponds to the DSD concept  $C_{dsd}$ .

Mapping can be applied with the two following purposes:

- a) Building an SQL Query,
- b) Creating an SDMX data file.

### 4.2.2 Building an SQL Query

Action is taken once an SQL query has been generated further to the process of parsing an SDMX Query Message. This SQL must then be mapped using the local dissemination database's concepts.

In the SDMX Query Message that is to be parsed, one finds the 'query:DataWhere' element. This element includes the constraints that are necessary to select the data inside the database. These constraints are represented by child elements inside the 'query:DataWhere' element, and can be joined with an 'And' or an 'Or' condition. Figure 7 contains an example of the 'query:DataWhere' element of an SDMX Query Message:

```
<query:DataWhere>
  <query:And>
    <query:Dimension id="Adsd">a1dsd</query:Dimension>
    <query:Dimension id="Bdsd">b1dsd</query:Dimension>
    <query:Time>
      <query:StartTime>2006-01</query:StartTime>
      <query:EndTime>2006-04</query:EndTime>
    </query:Time>
  <query:Or>
    <query:Dimension id="Cdsd">c1dsd</query:Dimension>
    <query:Dimension id="Cdsd">c2dsd</query:Dimension>
  </query:Or>
</query:And>
</query:DataWhere>
```

**Figure 7 - '<query:DataWhere>' element within an SDMX Query Message**

<sup>4</sup> Data defined by the data provider.

<sup>5</sup> The names, code lists and values of the local concepts differ from the names, code lists and values of the DSD concepts.

The result of parsing the ‘query:DataWhere’ is a ‘where clause’ of an SQL query. In Figure 7, for example, the elements and the parsing process are:

1. The first ‘query:And’ element. This implies that all of this element's child elements are related using an ‘and’ condition.
2. The ‘query:Dimension’ elements. They are given by the ‘id’ XML attribute of the Dimension XML elements. The dimensions are  $A_{dsd}$ ,  $B_{dsd}$  and  $C_{dsd}$ . The values given are those of each related element:

$$A_{dsd} = a1_{dsd}$$

$$B_{dsd} = b1_{dsd}$$

$$C_{dsd} = c1_{dsd}$$

$$C_{dsd} = c2_{dsd}$$

3. The ‘Time’ element. It contains the ‘StartTime’ and the ‘EndTime’ child elements, which respectively represent the beginning and end of the data observation period. The Boolean relationship between the two constraints is ‘and’. As a result:

$$\text{StartTime} = 2006-01 \text{ and } \text{EndTime} = 2006-04$$

4. The ‘query:Or’ element. This implies that all of this element's child elements are related using an ‘or’ condition. From the DSD, the result of the ‘query:Or’ element is:

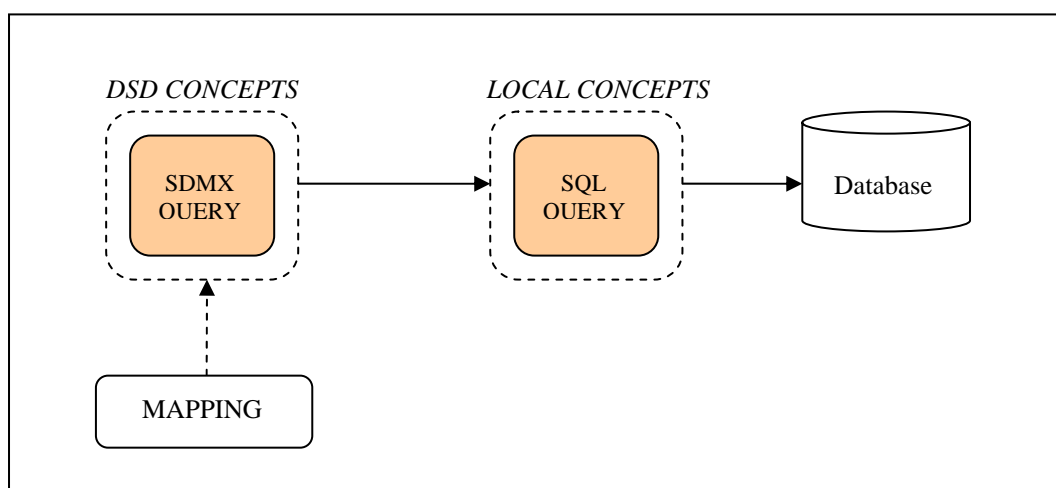
$$C_{dsd} = c1_{dsd} \text{ OR } C_{dsd} = c2_{dsd}$$

Once all the elements of the ‘query:DataWhere’ element have been parsed, the result is:

$$\text{“}A_{dsd} = a1_{dsd} \text{ AND } B_{dsd} = b1_{dsd} \text{ AND } (C_{dsd} = c1_{dsd} \text{ OR } C_{dsd} = c2_{dsd}) \text{ AND STARTTIME} = 2006-01 \text{ AND ENDTIME} = 2006-04\text{”}$$

The ‘where’ clause resulting from the SQL query is expressed using DSD concepts, whereas inside the database data are expressed using local concepts. This means that, in order to retrieve data, it becomes necessary to translate DSD concepts into local concepts. This is done by using the mapping process (compare Figure 8). In the example, using the  $T_{loc}$  concept, the result of the translation is:

$$\text{“}A_{loc} = a1_{loc} \text{ AND } B_{loc} = b1_{loc} \text{ AND } (C_{loc} = c1_{loc} \text{ OR } C_{loc} = c2_{loc}) \text{ AND } T_{loc} \text{ BETWEEN } 200601 \text{ AND } 200604\text{”}$$



**Figure 8 - Translation schema of an SDMX query**

#### 4.2.3 Creating an SDMX data file

The SDMX-ML data file must contain DSD concepts and codes, while the data retrieved by the dissemination database contains only local concepts and codes. The SDMX query used in the previous example is further used as an example of how to retrieve a compact SDMX data file.  $A_{dsd}$ ,  $B_{dsd}$  and  $C_{dsd}$  are the dimensions.

The dataset retrieved by the dissemination database is shown in Table 3:

$T_{loc}$	Data	$A_{loc}$	$B_{loc}$	$C_{loc}$
200601	84.5	$a1_{loc}$	$b1_{loc}$	$c1_{loc}$
200602	85.6	$a1_{loc}$	$b1_{loc}$	$c1_{loc}$
200603	95.7	$a1_{loc}$	$b1_{loc}$	$c1_{loc}$
200604	94.2	$a1_{loc}$	$b1_{loc}$	$c1_{loc}$
200601	82.1	$a1_{loc}$	$b1_{loc}$	$c2_{loc}$
200602	87.2	$a1_{loc}$	$b1_{loc}$	$c2_{loc}$
200603	88.0	$a1_{loc}$	$b1_{loc}$	$c2_{loc}$
200604	87.9	$a1_{loc}$	$b1_{loc}$	$c2_{loc}$

**Table 3 - Dataset retrieved**

In order to generate the SDMX-ML file, the retrieved data must be translated into DSD concepts (Figure 9):

$T_{dsd}$	$M_{dsd}$	$A_{dsd}$	$B_{dsd}$	$C_{dsd}$
200601	84.5	$a1_{dsd}$	$b1_{dsd}$	$c1_{dsd}$
200602	85.6	$a1_{dsd}$	$b1_{dsd}$	$c1_{dsd}$
200603	95.7	$a1_{dsd}$	$b1_{dsd}$	$c1_{dsd}$
200604	94.2	$a1_{dsd}$	$b1_{dsd}$	$c1_{dsd}$
200601	82.1	$a1_{dsd}$	$b1_{dsd}$	$c2_{dsd}$
200602	87.2	$a1_{dsd}$	$b1_{dsd}$	$c2_{dsd}$
200603	88.0	$a1_{dsd}$	$b1_{dsd}$	$c2_{dsd}$
200604	87.9	$a1_{dsd}$	$b1_{dsd}$	$c2_{loc}$

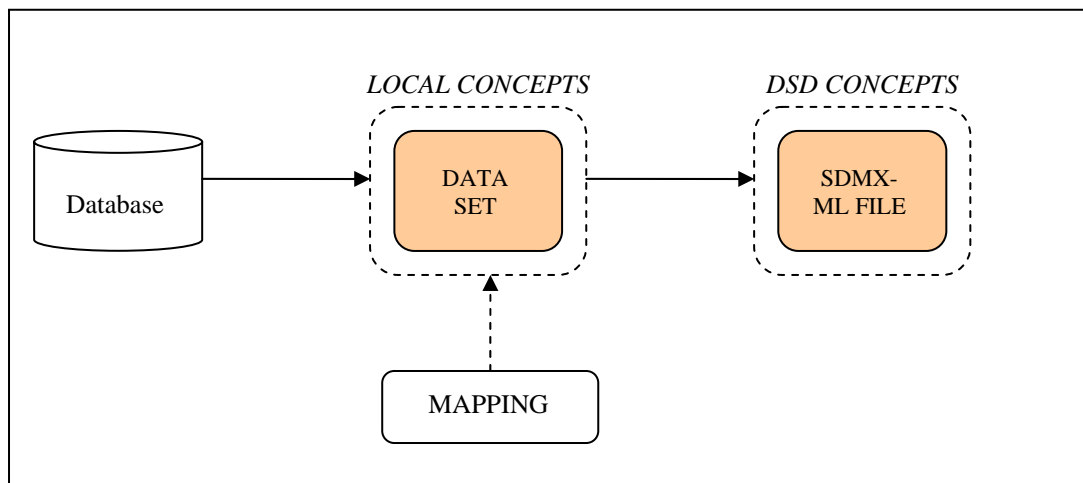
```

<estat_sts:DataSet>
  <estat_sts:Series Adsd="a1dsd" Bdsd="b1dsd" Cdsd="c1dsd" >
    <estat_sts:Obs Mdsd="84.5" Tdsd="2006-01" />
    <estat_sts:Obs Mdsd="85.6" Tdsd="2006-02" />
    <estat_sts:Obs Mdsd="95.7" Tdsd="2006-03" />
    <estat_sts:Obs Mdsd="94.2" Tdsd="2006-04" />
  </estat_sts:Series>
  <estat_sts:Series Adsd="a1dsd" Bdsd="b1dsd" Cdsd="c2dsd" >
    <estat_sts:Obs Mdsd="82.1" Tdsd="2006-01" />
    <estat_sts:Obs Mdsd="87.2" Tdsd="2006-02" />
    <estat_sts:Obs Mdsd="88.0" Tdsd="2006-03" />
    <estat_sts:Obs Mdsd="87.9" Tdsd="2006-04" />
  </estat_sts:Series>
</estat_sts:DataSet>

```

**Figure 9 - Translation of dataset into SDMX-ML file**

For this translation the following schema applies (Figure 10):



**Figure 10 - Translation schema of a dataset**

### 4.3 The mapping process – example: CENSUS HUB

A data example from the Census hub project is used to explain the mapping process.



### 4.3.1 Step 1: List of local concepts

The type of storage used in this database comprises more than one table (compare Figure 11). The primary table has one primary key called 'PK\_DATO'. The table is completed by the columns 'AGE\_CMP', 'CAS\_CMP', 'GEO\_CMP', 'SEX\_CMP', 'FK\_DATAFLOW' and 'FREQ'. The secondary table is joined to the first one by the primary key of the primary table. It also contains the columns 'OBS\_VALUE\_CMP' and 'TIME\_CMP'.

#### Primary table:

PK_DATO	AGE_CMP	CAS_CMP	GEO_CMP	SEX_CMP	FK_DATAFLOW	FREQ
1	TOT	TOT	X_CZ	T	CENSUSHYB_Q_XS1	1
10	TOT	TOT	X_CZ	T	CENSUSHYB_Q_XS1	1
100	TOT	002	X_CZ	T	CENSUSHYB_Q_XS1	1
1000	003	001	X_CZ	T	CENSUSHYB_Q_XS1	1
89	003	001	X_CZ	T	CENSUSHYB_Q_XS1	1

#### Secondary Table:

PK_DATA	OBS_VALUE_CMP	TIME_CMP
100	666869	2001Y
89	26727	2001Y

**Figure 11 - Local Database Storage (Two Tables)**

The list of local concepts is shown in the Table 4:

Local columns	Description
FREQ	Frequency of series - Represents the frequency at which data are produced. In this database the value 'A' indicates that the frequency is annually
AGE_CMP	Age groups - these represent the ages levels at which information is grouped.
CAS_CMP	Activity status groups - defines the activity status levels at which information is grouped.
GEO_CMP	Geographical groups - Represents the classification of geographical groups.
SEX_CMP	Sex groups - Represent the sex groups' classification.
FK_DATAFLOW	This is the foreign key of the dataflow it belongs to.
OBS_VALUE_CMP	The statistical value.
TIME_CMP	The time of the observation value (the year in this case)

**Table 4 - List of local concepts**

#### 4.3.2 Step 2: Associating all local concepts with DSD concepts.

Continuing with the database example introduced in step 1, 'CENSUSHUB+ESTAT+2.0' DSD is taken as an example to describe the association process of DSD concepts and local database concepts. The concepts used are shown in Table 5:

Id	Description	Type of concept	Obligatory?
FREQ	Frequency	Dimension	
AGE	Age indicator	Dimension	
CAS	Activity status indicator	Dimension	
GEO	Geographical indicator	Dimension	
SEX	Sex indicator	Dimension	
TIME	Time period or range	TimeDimension	
OBS_STATUS	Observation status	Attribute	Not mandatory
TIME_FORMAT	Time format code	Attribute	Mandatory
OBS_VALUE	Observation value	Measure	

**Table 5 - DSD Concepts**

The names of the corresponding SDMX code lists and characteristic values used in the example are listed below in Table 6:

<b>Id</b>	<b>Code list used</b>	<b>Code(s) used</b>	<b>Description of code(s)</b>
FREQ	CL_FREQ	A	Annually
AGE	CL_AGE	TOT	Total
FREQ	CL_FREQ	A	Annually
OBS_STATUS	CL_OBS_STATUS	A	Normal value
CAS	CL_CAS	TOT	Total
		001	Persons aged 15 years and over
		002	Economically active
GEO	CL_GEO	CZ	Czech Republic
SEX	CL_SEX	TOT	Total
TIME	-	-	-
OBS_STATUS	CL_OBS_STATUS	A	Normal value
TIME_FORMAT	CL_TIME_FORMAT	P1Y	Annually
OBS_VALUE	-	-	-

**Table 6 - DSD Concepts – Code lists and codes**

The following actions are performed in the mapping process and summarised in Table 7:

- The local concept 'AGE\_CMP' represents the age indicator. This means that it can be associated to the DSD concept 'AGE', which also represents the age indicator inside the DSD structure.
- The local concept 'CAS\_CMP' can be associated with the DSD concept 'CAS'.
- 'GEO\_CMP' can be associated with the DSD concept 'GEO'.
- The local concept 'FREQ' can be associated to DSD concept 'FREQ'.
- 'SEX\_CMP' can be associated with the DSD concept 'SEX'.
- The local concept 'TEMP\_CMP' can be associated to the DSD concept 'TIME'.
- The local concept 'OBS\_VALUE\_CMP' can be associated to the DSD concept 'OBS\_VALUE'.

LOCAL CONCEPT	DSD CONCEPT
FREQ	FREQ
AGE_CMP	AGE
CAS_CMP	CAS
GEO_CMP	GEO
SEX_CMP	SEX
TIME_CMP	TIME
-	TIME_FORMAT
-	OBS_STATUS
OBS_VALUE_CMP	OBS_VALUE

**Table 7 - Associations between local concepts and DSD concepts**

This mapping process between local concepts and DSD concepts is not always one-to-one. In the example introduced, the possibilities are:

- One local concept corresponds to one DSD concept:
  - Local concept 'FREQ' with DSD concept 'FREQ';
  - Local concept 'AGE\_CMP' with DSD concept 'AGE';
  - Local concept 'CAS\_CMP' with DSD concept 'CAS';
  - Local concept 'GEO\_CMP' with DSD concept 'GEO';
  - Local concept 'SEX\_CMP' with DSD concept 'SEX';
  - Local concept 'TIME\_CMP' with DSD concept 'TIME';
  - Local concept 'OBS\_VALUE\_CMP' with DSD concept 'OBS\_VALUE'.
- A DSD concept does not correspond to any local concept:
  - In the case of 'TIME\_FORMAT' and 'OBS\_STATUS', a mapping item needs to be added for these local database concepts because they are mandatory in the SDMX data file. They can be created using knowledge of the DSD domain and SDMX syntax. In the example, the concept 'TIME\_FORMAT' can be deduced from the DSD 'FREQ' concept. The 'OBS\_STATUS' is equal to 'A'='Normal Value';
  - The obtained values are used in the mapping of codes. This process is presented in Step 3: 'Mapping of codes'.

### 4.3.3 Step 3: Mapping codes

In order to better understand the mapping of codes, it is necessary to have a description of the different values the concepts can have in the example:

- In the local concept 'AGE\_CMP' code list, for example, 'TOT' corresponds to 'TOT' inside the DSD code list 'CL\_AGE'. The other values are also congruent, thus no mapping is required;
- The same applies for 'CAS\_CMP' and 'CAS', where both code lists are also congruent;
- In the local concept 'GEO\_CMP' code list, the code 'X\_CZ' would corresponds to the code 'CZ' in the DSD code list 'CL\_GEO'. Thus, code mapping – the so-called 'transcoding' – has to be performed;
- The same need for transcoding applies for the code lists/codes of the local concept 'SEX\_CMP' versus the DSD concept 'SEX';
- For the local concept 'FREQ', the code 'A' corresponds to the code 'A' of the DSD code list for annual and no transcoding is required;
- The DSD concepts TIME\_FORMAT and OBS\_STATUS do not have any corresponding local concept. A constant value 'P1Y' is kept as code for the mandatory concept TIME\_FORMAT during the mapping.

Examples of the resulting mapping of local codes with DSD codes are presented in Table 8:

LOCAL CONCEPT	LOCAL CODE	DSD CODE	DSD CONCEPT
Freq	A	A	FREQ
AGE_CMP	TOT	TOT	AGE
CAS_CMP	TOT 001 002	TOT 001 002	CAS
GEO_CMP	X_CZ X_IT	CZ IT	GEO
SEX_CMP	T M F	TOT 001 002	SEX
TIME_CMP	2001Y	2001	TIME
		P1Y	TIME_FORMAT
OBS_VALUE_CMP			OBS_VALUE
			OBS_STATUS

**Table 8 - Mapping between local code and DSD code**

**Once the mapping has been created, it can be used:**

1. To translate an SDMX query.
2. To translate a dataset.

#### 4.3.4 Translation of an SDMX query

Figure 12 contains an example of an SDMX Query that includes a ‘query:DataWhere’ element:

```
<query:DataWhere>
  <query:And>
    <query:Dimension id="AGE">TOT</query:Dimension>
    <query:Dimension id="GEO">CZ</query:Dimension>
    <query:Dimension id="SEX">TOT</query:Dimension>
    <query:Time>
      <query:Time>2001</query:Time>
    </query:Time>
    <query:DataProvider>IT1</query:DataProvider>
    <query>Dataflow>SSTSIND_PROD_M</query>Dataflow>
    <query:Or>
      <query:Dimension id="CAS">TOT</query:Dimension>
      <query:Dimension id="CAS">001</query:Dimension>
      <query:Dimension id="CAS">002</query:Dimension>
    </query:Or>
  </query:And>
</query:DataWhere>
```

**Figure 12 - Example of SDMX query**

Parsing the SDMX query yields the following 'where' clause inside the SQL query:

```
"AGE = 'TOT' AND GEO = 'CZ' AND SEX='TOT' AND (CAS = 'TOT' OR CAS =
'001' OR CAS = '002') AND TIME = 2001"
```

Applying the mapping, the translation of the DSD elements to the local elements that were already mapped is:

```
"AGE_CMP= 'TOT' and GEO_CMP='X_CZ' and SEX_CMP='T' and (CAS_CMP=
'TOT' or CAS_CMP='001' or CAS_CMP='002') and TIME_CMP='2001Y'"
```

Where:

- GEO = 'CZ' is equal to GEO\_CMP='X\_CZ'
- SEX = 'TOT' is equal to SEX\_CMP='T'
- TIME = 2001 is equal to TIME\_CMP='2001Y'

### 4.3.5 Translation to a dataset

The result of the previous query provides the following result set expressed in local concepts:

**Primary table:**

AGE_CMP	CAS_CMP	GEO_CMP	SEX_CMP	OBS_VALUE_CMP	TIME_CMP	FREQ
TOT	002	X_CZ	T	666869	2001Y	A

**Table 9 - Result set expressed in local concepts**

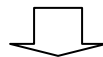
The local concepts and local codes are translated to the mapped values of the DSD:



TIME	OBS_VALUE	AGE	CAS	GEO	FREQ	SEX	OBS_STATUS	TIME_FORMAT
2001	666869	TOT	002	CZ	A	TOT	A	P1Y

**Table 10 - Dataset expressed in DSD concepts**

Once the mapping is finalized, the SDMX-ML data file (**Error! Reference source not found.**) can be created using the data from the dataset expressed in DSD concepts (Table 10).



```

<DataSet>
  < Series FREQ="A" AGE="TOT" CAS="002" GEO="CZ" SEX="TOT"
    TIME_FORMAT="P1Y">
    < Obs TIME ="2001" OBS_VALUE="666869" OBS_STATUS="A" />
  </ Series>
</DataSet>

```

**Table 11 presents the list of concepts and acronyms with their definition.**

## 5 **Glossary**

Concept	Definition
<i>CVS</i>	Comma-separated values
<i>DSD</i>	Data Structure Definition
<i>EDIFACT</i>	Electronic Data Interchange for Administration, Commerce and Transport
<i>ESMS</i>	Euro SDMX Metadata Structure
<i>GESMES/TS</i>	GESMES Time Series data exchange message
<i>HTTP</i>	Hypertext Transfer Protocol
<i>ISO</i>	International Organisation for Standardisation
<i>ISTAT</i>	National Statistical Institute of Italy
<i>IT</i>	Information Technology
<i>MSD</i>	Metadata Structure Definition
<i>NSI</i>	National Statistical Institution
<i>RSS</i>	Really Simple Syndication ( <i>also used Rich Site Summary</i> ) - family of Web feed formats to publish frequently updated information
<i>SDMX</i>	Statistical Data and Metadata eXchange.
<i>SDMX-EDI</i>	SDMX Electronic Data Interchange - EDIFACT format for exchange of SDMX-structured data and metadata
<i>SDMX-IM</i>	SDMX Information Model
<i>SDMX-ML</i>	SDMX Markup Language - XML format for the exchange of SDMX-structured data and metadata
<i>SOAP</i>	Simple Object Access Protocol
<i>SQL</i>	Sequence Query Language
<i>UML</i>	Unified Modelling Language
<i>URL</i>	Uniform Resource Locator
<i>XLS</i>	Excel Worksheet
<i>XML</i>	EXtensible Markup Language

**Table 11 – Glossary**