CCSDS DEDSL
A WAY TO DESCRIBE METADATA

Arnaud LUCAS (Arnaud.Lucas@cnes.fr)

Denis MINGUILLON (Denis.Minguillon@cnes.fr)

INTRODUCTION

The DEDSL provides an extensible way of defining data entity dictionaries.

A Data Entity is a concept that can, or does, take on one or more values. Semantics of a data entity, such as a text definition of its meaning, are defined by attributes. The purpose of the DEDSL Recommendation is to define a language for specifying a dictionary which describes semantics for a collection of data entities—it does not define a specific dictionary.

A dictionary is understood as a mechanism that is able to organise a set of information in a consistent and easily understandable manner, and it is commonly used by humans to look up the meaning of words used in natural languages. Similarly, a Data Entity Dictionary (DED) is used by humans and systems to look up the definition, and other attributes, of data entities used in the definition and generation of data products.

The DEDSL defines the abstract definition of the semantic information that is required to be conveyed and presents the specification in a layered manner (attributes, entities, dictionaries). This is done so that the actual technique used to convey the information is independent of the information content and, therefore, the same abstract standard can be used within different formatting environments. This also permits the semantic information to be translated to different representations as may be needed when data are transferred across different domains.

The DEDSL Recommendation defines the concepts of name, definition, units, and a small set of other standard attributes so that they may be used consistently in the formation of data entity dictionaries. Given the wide variety of data entities that may need to be described, only a few of the attributes are made mandatory by this Recommendation.

The method used to define standard attributes can also be used to extend the set of attributes beyond the standard ones provided within the Recommendation.

Several classes of data entities are defined. These classes allow making a distinction between the abstract data entities—the models—and the concrete data entities—the data fields in a data product.

This Recommendation is strongly based on the International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) Specification and standardisation of data elements (*Information Technology—Specification and standardisation of data elements—part 3: Basic attributes of data elements*. ISO/IEC 11179-3:1994. Geneva: ISO, 1994) with which it is widely consistent for all semantic aspects.

OVERVIEW

GENERAL

As discussed in the introduction, a Data Entity Dictionary is used by humans and systems to look up the definition and other attributes of data entities. This section discusses some of the primary uses of data entity dictionaries and presents a basic example of DEDSL usage.

USES OF DATA ENTITY DICTIONARIES

PRODUCT DATA ENTITY DICTIONARIES

So that its contained data can be extracted, a data product can appear with a formatting standard (e.g., Flexible Image Transport System [FITS]), or a self-describing format (e.g., CDF, HDF, etc.) or a Data Definition Language (e.g., EAST). This syntactic information may not be easy to understand and a formal definition of additional semantics may be necessary, which leads to the definition of a product Data Entity Dictionary.

Figure -1- shows the structure of the data product (**PRODUCT_X**). It is made up of a header (**HEADER**) and an image (**DATA_1**). The header contains a product identifier (**PRODUCT_ID**), information about the station, which has acquired the data, (**ACQ_STATION**), information about the acquisition time (**ACQ_TIME**), and information necessary for the processing of the image, e.g., the centre coordinates [**CENTRE_COORD** (**LATITUDE** and **LONGITUDE**)]. Its physical description, possibly expressed using a DDL, may not be readily understandable to all readers because it includes a specification to the bit level. It is also useful to have a quick overview of the product and to have additional semantic information. Therefore, the definition of a product data entity dictionary (**PRODUCT_X DED**) will bring this necessary information and perspective.

Within this data entity dictionary additional information can be given to more precisely define:

– the definition of each data entity;

– the kinds of values used for the centre coordinates **LATITUDE** and **LONGITUDE**.

The values of **LATITUDE** are defined relative to the Equator and range from -90.000 to +90.000, while the values of **LONGITUDE** are defined to be relative to Greenwich and range from -180.00 to +180.00. The units used to express the values are to be in degrees. All this information may be included in the data entity dictionary.

These pieces of semantic information correspond to the existing data entities of the product.

Consequently, the **PRODUCT_X DED** will contain the semantic descriptions of the following data entities:

- HEADER;
- PRODUCT_ID;
- ACQ_STATION;
- ACQ_TIME;
- CENTRE_COORD;
- LATITUDE;
- LONGITUDE;
- DATA_1.

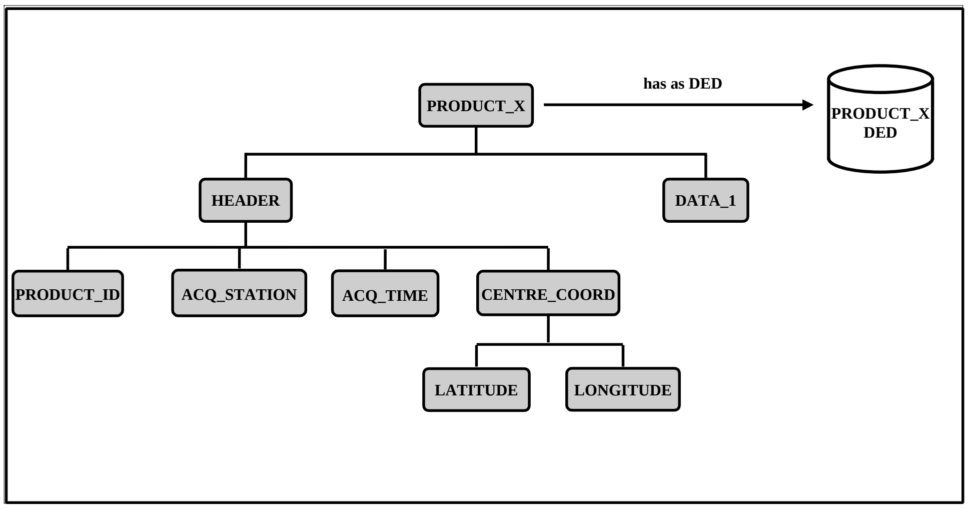


Figure 1: Organisation of the Data Product Product_X

Figure -2- shows the structure of another data product (**PRODUCT_Y**). It is made up of the product identifier (**PRODUCT_ID**), the centre coordinates (e.g., a latitude, a longitude) and the acquired image (**DATA_2**). A data product dictionary (**PRODUCT_Y DED**) can also be defined for **PRODUCT_Y** in the same way as for **PRODUCT_X**.

Consequently, the **PRODUCT_Y DED** will contain the semantic descriptions of the following data entities:

- PRODUCT_ID;
- LATITUDE;
- LONGITUDE;
- DATA_2.

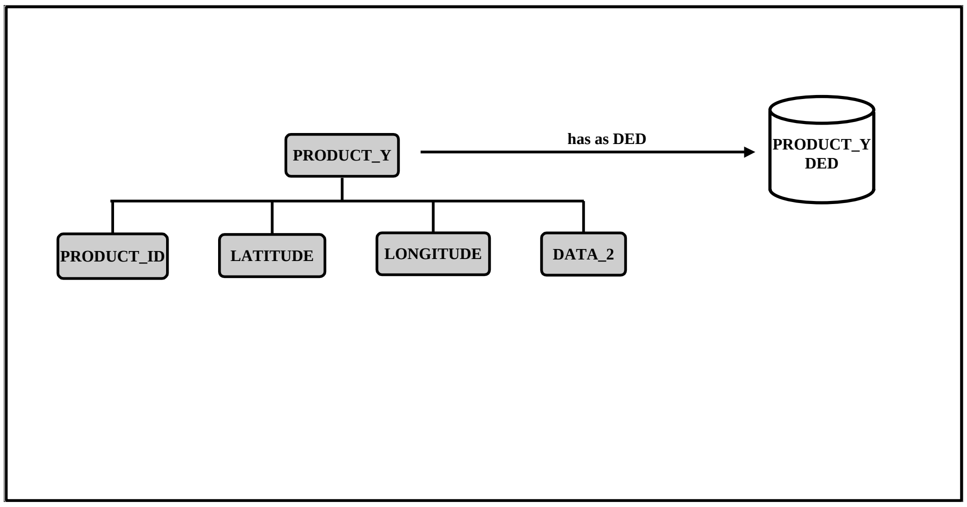


Figure 2: Organisation of the Data Product Product_Y

Looking at the two previously described data products it seems convenient to define **PRODUCT_Y DED** by re-using some data entity descriptions of **PRODUCT_X DED.**

The data entity descriptions which seem common to both data products are **PRODUCT_ID, LATITUDE** and **LONGITUDE**.

Therefore the data product dictionary **PRODUCT_X** can be modified so that those data entity descriptions become re-usable **models**. These models are abstract data descriptions to which concrete descriptions, i.e., corresponding to data entities within the data product, can refer. Then the DED associated with **PRODUCT_Y** can refer to the DED associated with **PRODUCT_X** for the definition of some of its semantic descriptions using the models of the **PRODUCT_X DED.**

Consequently, the **PRODUCT_X DED** will contain data entity descriptions corresponding to abstract definitions (models) and named as follows:

- PRODUCT_ID_MODEL;
- LATITUDE_MODEL;
- LONGITUDE_MODEL.

The **PRODUCT_X DED** will still contain the semantic descriptions corresponding to the following data entities, but with references to the newly defined models:

- HEADER;

- PRODUCT_ID, inheriting from the model PRODUCT_ID_MODEL;
- ACQ_STATION;
- ACQ_TIME;
- CENTRE_COORD;
- LATITUDE, inheriting from the model LATITUDE_MODEL;
- LONGITUDE, inheriting from the model LONGITUDE_MODEL;
- DATA_1.

The **PRODUCT_Y DED** will still contain the semantic descriptions corresponding to the following data entities, but with references to the newly defined models:

- PRODUCT_ID, inheriting from the model PRODUCT_ID_MODEL;
- LATITUDE, inheriting from the model LATITUDE_MODEL;
- LONGITUDE, inheriting from the model LONGITUDE_MODEL;
- DATA_2.

Figure -3- presents the resulting organisation between both DEDs as they are used to support PRODUCT_Y.

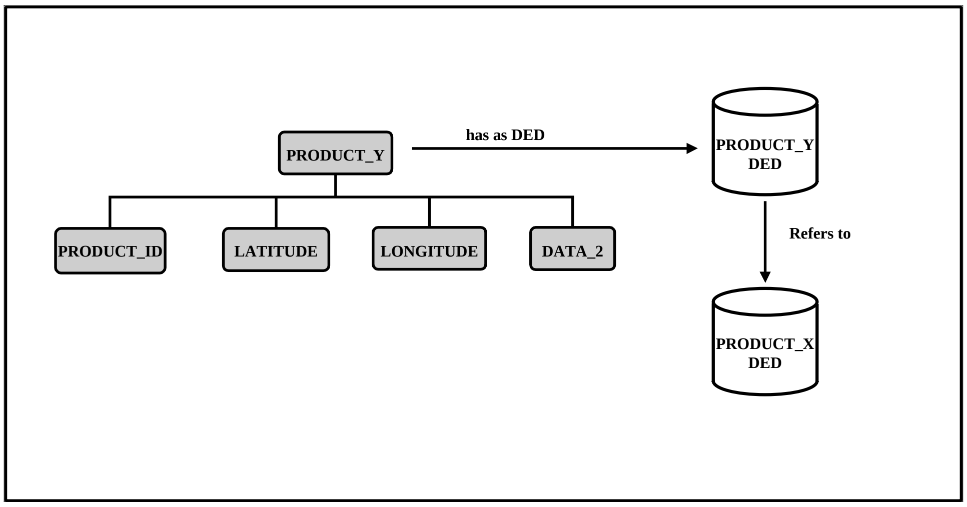


Figure 3: Organisation of the DED Relative to Product_X and Product_Y

COMMUNITY DATA ENTITY DICTIONARIES

The project or the data designer may consider that a community dictionary is necessary because there are several data products related to the same kind of data.

Looking at the two products given as examples, they may decide that the community data entity dictionary should include the following entities which frequently appear: **PRODUCT_ID**, **ACQ_STATION**, **ACQ_TIME, LATITUDE** and **LONGITUDE**.

This community DED will contain a normalized description of these entities which can then be considered as **models**. Data entities being latitudes and longitudes and appearing within other data products will then have the same associated semantic information whenever they inherit from these normalized descriptions.

The other data entities only appearing in a data product such as **`DATA_1, DATA_2`** and **`CENTRE_COORD`** are local definitions.

The purpose of a community dictionary is to provide, across different data products, a standard or normalized definition of data entities.

Table -1- shows an example of mapping of the community DED entries into data product DED entries for the data products shown in figures -1- and -2-, according to the choices made by the project or data designers.

Table 1: Comparison of Community DED and Product DED Descriptions

Concept	**Data Type**	**Found in Community DED**	**Found in PRODUCT_X DED**	**Found in PRODUCT_Y DED**
HEADER	Composite	no	yes	no
PRODUCT_ID	Text	yes	yes	yes
ACQ_STATION	Enumeration	yes	yes	no
ACQ_TIME	Composite	yes	yes	no
CENTRE_COORD	Composite	no	yes	no
LATITUDE	Real	yes	yes	yes
LONGITUDE	Real	yes	yes	yes
DATA 1	Composite (Array of 16-bit integers)	no	yes	no
DATA 2	Composite (Array of real numbers)	no	no	yes

Supposing that the project has defined its community dictionary, when it defines a new data product or for example when it rewrites the dictionary related to **`PRODUCT_X`** (in figure -1-), it can decide to define the **`HEADER`** entity on the basis of **`PRODUCT_ID`**, **`ACQ_STATION`** and **`ACQ_TIME`**, which inherit from the corresponding data entities in the community dictionary, i.e., which then have the same properties as the model data entities. The project can also decide to define the **`CENTRE_COORD`** entity on the basis of the **`LATITUDE`** and **`LONGITUDE`** entities, which inherit from the corresponding data entities in the community dictionary. The **`DATA1`** entity does not inherit from a specific data entity described in the community DED, as it appears to be a kind of data only appearing in the data product **`PRODUCT_X`**.

The same policy can be applied to **`PRODUCT_Y`** of figure -2-.

Therefore we can consider the links 'uses some models of', in figure -4-, among the three data entity dictionaries. A 'uses some models of' link between **`PRODUCT_X DED`** and the domain DED means that some data entities of **`PRODUCT_X`** inherit from corresponding data entities contained in the domain DED.

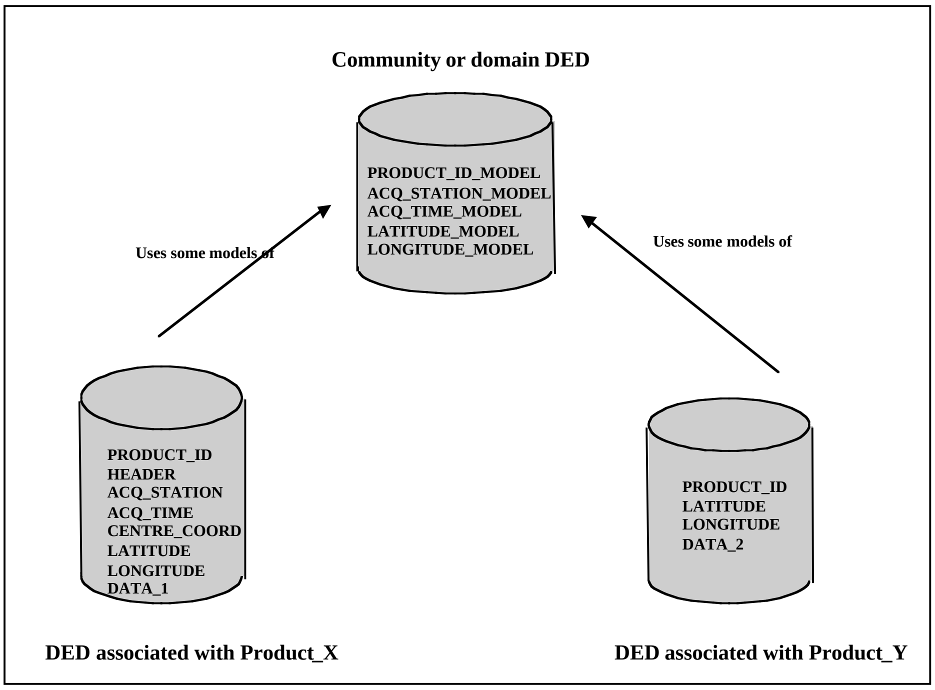


Figure 4: 'Uses some models of' Links between Data Product Dictionaries

APPLICATION OF THE DEDSL

GENERAL

As demonstrated in the previous section, there are two major uses for dictionaries:

– to describe a data product semantically;

– to build up and define a community DED.

The Recommendation focuses on developing standard names and descriptions for the concepts required for Data Entity Dictionaries, and formally defines the concepts of name, definition, units, and a small set of other attributes so they may be used consistently in the formation of data entity dictionaries. A method is also provided to permit the set of attributes to be extended beyond the standard ones provided within this Recommendation.

This formal definition enables the definition of generic tools to assist producers in creating documented products, and to assist consumers in understanding the products they receive.

PRODUCT DATA ENTITY DICTIONARIES

A product DED is a means for an organisation to present the semantics necessary for a good understanding of a managed data product. A product DED is dependent of the data product description. However, an enhanced understanding is only possible when the semantics associated with the products are presented in a common, i.e., standardised way.

Therefore, the DEDSL Recommendation provides a foundation for the creation of a product DED by providing a basic set of concepts for data entity descriptions. This

Recommendation also provides the formal methods to describe relationships among the data entities of the product DEDs.

COMMUNITY DATA ENTITY DICTIONARIES

A community DED is also a means for an organisation to gain some degree of control/standardisation over the data descriptions created by member data producers. Unlike product data entity dictionaries, these community DED are not used in conjunction with a specific product description technique. They are independent of the specific implementation of products. Examples of uses of community DED include:

– the creation of a standard data entity dictionary by an organisation that mandates the attributes defining each entity description in dictionaries used within that organisation;

– the creation of a community DED by a particular community (e.g., planetary science, astrophysics, etc.), to establish a degree of standardisation for the contents of any data entity dictionary associated with a data product from that community.

The DEDSL Recommendation provides a foundation for the creation of community DED and also provides the formal methods to describe relationships among data entities of multiple data entity dictionaries.

REGISTERING DED

Whenever a project or data designer has defined a product DED, it makes sense to register it as it may apply to multiple instances of the product and this makes it easier to find and retrieve for dissemination or updating.

Whenever a project or data designer has defined a community DED, they can decide to register it at different levels:

– in the framework of any organisation dealing with data of a particular domain or project;

– internally within an agency;

– within the CCSDS community.

For example, whenever a member agency considers that one of its particular community DED corresponds to the needs of other agencies, it may submit its DEDs to an organisation conforming to the CCSDS (references [4] and [C5]) or ISO registration procedures.

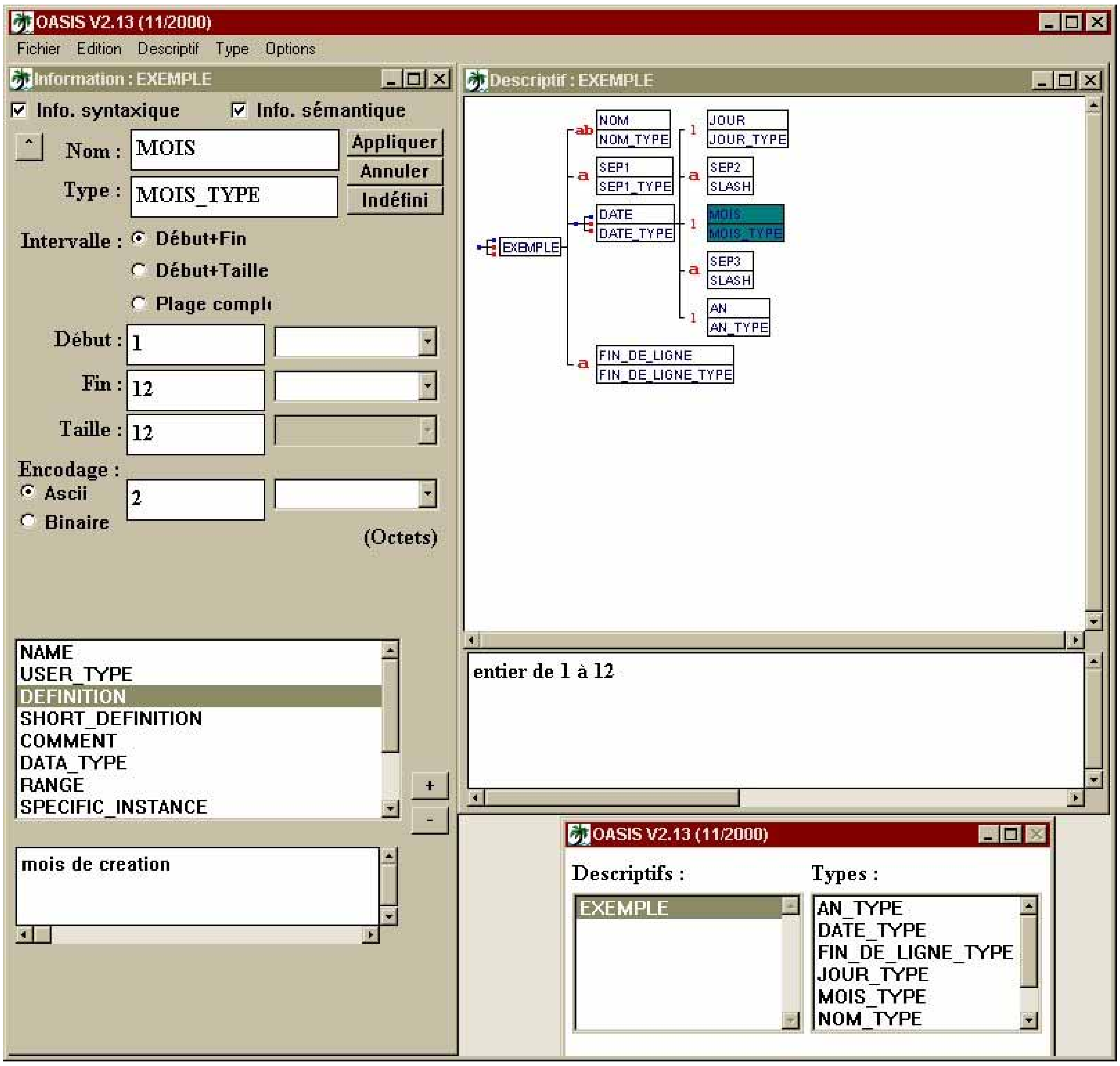
IMPLEMENTATIONS

There are currently two implementations of the DEDSL :

The PVL implementation - **CCSDS 647.2-B-1**: *Data Entity Dictionary Specification Language (DEDSL) - PVL Syntax (CCSD0012).*

The XML implementation - **CCSDS 647.3-B-1**: *Data Entity Dictionary Specification Language (DEDSL) - XML/DTD Syntax (CCSD0013).*

Both are implemented in the CNES Data Description Tool : OASIS (http://east.cnes.fr). OASIS allows the production of Data Entities Dictionaries through a user-friendly graphical interface. DEDs in XML format are used by several project at CNES, to produce interface documentation. Figure -5- shows a snapshot of the OASIS screen.



[✓] Info. syntaxique [✓] Info. sémantique

Nom : MOIS

Type : MOIS_TYPE

Intervalle : [✓ Début+Fin] Début+Taille Plage compl

Début : 1

Fin : 12

Taille : 12

Encodage : [✓ Ascii] Binaire 2 (Octets)

NAME
USER_TYPE
DEFINITION
SHORT_DEFINITION
COMMENT
DATA_TYPE
RANGE
SPECIFIC_INSTANCE

mois de creation

entier de 1 à 12

OASIS V2.13 (11/2000)

Descriptifs : EXEMPLE

Types : AN_TYPE, DATE_TYPE, FIN_DE_LIGNE_TYPE, JOUR_TYPE, MOIS_TYPE, NOM_TYPE

Figure 5: Screen of the CNES OASIS Tool

SUMMARY SESSION 4