Earth Observation Payload Data Ground Systems Infrastructure Evolution 2011-2014

LTDP SAFE

Corrupted Data Handling using DFDL Trade-Off

Ref: PDGS-SAFE-GMV-TN-12/0183

Version: 1.0

Date: 17th October 2012

Author	Miguel Vieira, GMV	17-10-2012 X M. Vieira Signed by: Carlos Miguel Lopes de Oliveira Vieira
Reviewer	Adrián Sanz, GMV	17/10/2012 X A. Sanz GMV SAFE Project Manager Firmado por: Adrián Sanz Díaz
Approver		

The work described in this document was performed under ESA Contract.
Responsibility for the contents resides in the author or organisation that prepared it.

ESA UNCLASSIFIED - For Official Use

Table of Contents

List of Tables

1 Introduction

1.1 Purpose

This document provides the trade-off of using DFDL to specify data quality information. The purpose of this document is to analyse the DFDL standard and verify if it can be used for representing data corruption in a SAFE package.

1.2 Scope

The scope of this document is to analyse the possibility of using DFDL to represent corrupted data elements within SAFE. This analysis has been requested as results of the PDR-C Actions (A03) within the LTDP-SAFE project. The results obtained from this analysis will be used as a basis to update the SAFE core documentation.

1.3 Document Status

This is the first version of the document issued for open discussion in the SAFE Wiki/Forum web page (http://wiki.services.eoportal.org/tiki-index.php?page=LTDP+SAFE+Wiki).

1.4 Applicable Documents

The following table lists the Applicable Documents that have a direct impact on the contents of this document.

Acronym	Title	Reference	Issue
[SAFE-P]	Standard Archive Format for Europe - Primer	PGSI-GSEG-EOPG-FS-010-0001	1.14
[SAFE-CB]	Standard Archive Format for Europe – Control book, Core Specifications	PGSI-GSEG-EOPG-FS-05-0001	1.14
[SAFE-RS]	Standard Archive Format for Europe – Recommendation for Specializations	PGSI-GSEG-EOPG-FS-05-0002	1.14
[DFDL]	Data Format Description Language (DFDL) v1.0 Specification	SpecificationGFD_174	1.0
[EOMPM]	Earth Observation Metadata profile of Observations & Measurements	OGC 10-157r3	1.0
[MDTO]	Metadata Definition Trade-Off	PDGS-SAFE-GMV-TN-12-0082	1.0

Table 1-: Applicable Documents

1.5 Reference Documents

Acronym	Title	Reference	Issue
	N/A		

Table 1-: Reference Documents

1.6 Acronyms and Abbreviations

The acronyms used in this document whose meaning has been deemed to be important are listed below. Additional acronyms may be found at [SAFE-P].

Acronym	Meaning
XML	eXtensible Markup Language
DFDL	Data Format Description Language
XSL	EXtensible Stylesheet Language
XSLT	EXtensible Stylesheet Language Transformations
EO	Earth Observation

Table 1-: Acronyms

2 Data Quality Representation in SAFE

Data quality can be affected by missing or corrupted data.

In a SAFE package the existence of missing/corrupted data can occur for several reasons, but if detected, the inclusion of information regarding the existing problems helps to mitigate the impact of data quality problems.

Data corruption refers to errors in data that can occur during writing, reading, storage, transmission, or processing, which introduces unintended changes. In data corruption, the corrupted units are present in the product but their value is wrong.

Missing data is a significantly different problem; the missing units are not present in the product.

While it is possible for corrupted units to be located (e.g. using an XPath), it's impossible to locate the missing units since they do not exist in the product.

The existing version (1.3) of SAFE has specific elements to handle missing data - `safe:missingElements`. DFDL is intended to specify information (text or binary) and with missing elements there is no information to specify. DFDL only allows specifying an applicable default value to a missing element in the data stream, i.e., DFDL allows specifying what to do when an element is missing, but not that an element is missing.

Missing data is out of the scope of this document, therefore only corrupted data will be addressed.

In SAFE format v1.3 specification ([SAFE-P] and [SAFE-CB]) lists the different possible situations in data quality problems and the approach to identify each of these cases:

2.1 Corrupted data cases:

2.1.1 Unknown Location

2.1.1.1 Corrupted units have been detected, but their location is completely unknown

```
<safe:corruptedElements>
    <safe:location>
        <safe:path>
            measurement/record/data
        </safe:path>
    </safe:location>
    <safe:count value="3"/>
</safe:corruptedElements>
```

Example 2-: 3 Detected Corrupted Data Elements description

2.1.1.2 Corrupted units have been detected, but beginning of the product is known and correct

Unknown Location (but it is known that the records #1 to #100 are correct)

```
<safe:corruptedElements>
    <safe:location>
        <safe:path>
            <![CDATA[measurement/record[fn:position() > 100]/data]]>
        </safe:path>
    </safe:location>
    <safe:count value="3"/>
</safe:corruptedElements>
```

Example 2-: 3 Detected Corrupted Data Elements description

2.1.1.3 Corrupted units have been detected, but the end of the product is known and correct

Unknown Location (but it is known that the records #101 until the end are correct)

```
<safe:corruptedElements>
    <safe:location>
        <safe:path>
<![CDATA[measurement/record[fn:position() < 101]/data]]>
        </safe:path>
    </safe:location>
    <safe:count value="3"/>
</safe:corruptedElements>
```

Example 2-: 3 Detected Corrupted Data Elements description

2.1.2 Location is unknown

2.1.2.1 Corrupted units have been detected in a segment of units

Data of records between #200 and #300

```
<safe:corruptedElements>
    <safe:location>
        <safe:path>
<![CDATA[measurement/record[fn:position()   >   200   and   fn:position()   <
300]/data]]>
        </safe:path>
    </safe:location>
    <safe:count value="3"/> non mandatory
</safe:corruptedElements>
```

Example 2-: 3 Detected Corrupted data elements

2.1.3 Location is known

2.1.3.1 Corrupted units have been detected and located (neither first nor last elements of a sequence)

```
<safe:corruptedElements>
    <safe:location>
        <safe:path>
<![CDATA[measurement/record[fn:position() > 3 and fn:position() < 7 ]/data]]>
        </safe:path>
    </safe:location>
    <safe:count value="3"/> non mandatory
</safe:corruptedElements>
</safe:corruptedElements>
```

Example 2-: 3 Detected Corrupted data elements (data of records #4, #5, #6)

2.1.3.2 Corrupted units (first elements of a sequence) have been detected

Data of records #1, #2, #3

```
<safe:corruptedElements>
    <safe:location>
        <safe:path>
            <![CDATA[measurement/record[fn:position() < 4]/data]]>
        </safe:path>
    </safe:location>
    <safe:count value="3"/> non mandatory
</safe:corruptedElements>
</safe:corruptedElements>
```

Example 2-: 3 Detected Corrupted data elements

2.1.3.3 Corrupted units (last elements of a sequence) have been detected

Data of records #100, #101, #102 - the product last record is the #102

```
<safe:corruptedElements>
    <safe:location>
        <safe:path>
            <![CDATA[measurement/record[fn:position() > 99]/data]]>
        </safe:path>
    </safe:location>
    <safe:count value="3"/> non mandatory
</safe:corruptedElements>
```

Example 2-: 3 Detected Corrupted data elements

2.1.3.4 Corrupted units have been detected and time located

Between 2004-12-16T23:55:53.000000Z and 2004-12-16T23:55:57.55555Z

```
<safe:corruptedElements>
    <safe:location>
        <safe:time>
            <safe:start>2004-12-16T23:55:53.000000Z</safe:start>
            <safe:stop>2004-12-16T23:55:57.55555Z</safe:stop>
        </safe:time>
        <safe:path>
            measurement/record
        </safe:path>
    </safe:location>
</safe:corruptedElements>
```

Example 2-: 3 Detected Corrupted Records

3 DFDL as Alternative to Represent Corrupted Data

Data Format Description Language (DFDL) is a language for describing text and binary data formats.

Using DFDL, a XML schema is written for the logical model of the data and special DFDL annotations are used to describe the native (non-XML) format of the data.

This is approach is allegedly already being used today in commercial systems. DFDL evolves this approach into an open standard capable of describing almost any format of text or binary data.

DFDL is descriptive and not prescriptive. DFDL is not a data format, nor does it impose the use of any particular data format. Instead it provides a standard way of describing many different kinds of data format.

This approach allows designing an appropriate data representation according to requirements and describes it in a standard way which can be shared, enabling multiple programs to directly interchange the data.

The motivations for this approach are to avoid inventing a completely new schema language, and to make it easy to convert general text and binary data, via DFDL information set, into a corresponding XML document, while making sharing much easier as long as it's compliant with the standard.

When trying to address data quality information specifications with DFDL, there are issues that should be considered:

- Corrupted data location.

  One issue regarding the use of DFDL to describe corrupted data is related with the corrupted data location.

  DFDL allows aligning data location using bytes and bits, therefore there is no direct mapping between DFDL and data location available resources under SAFE or the level of detail it provides. For instance, DFDL does not allow the specification of data location using time based intervals like in

  `<safe:qualityInformationType>/<safe:location>/<safe:time>/<safe:start>`

  and

  `<safe:qualityInformationType>/<safe:location>/<safe:time>/<safe:stop>`

  that can be done with SAFE.

  To support time and unit based location specification, DFDL would have to be extended thus resulting in loosing DFDL standards compliance and therefore the advantages of using an open standard format.

- DFDL is a specification language.

  DFDL provides a way of specifying any data format. If DFDL is used to describe corrupted data, it will handle it like any other data. DFDL allows data to be discarded or ignored, but not identifying it as corrupted. If discarding data is acceptable when handling corrupted

data, DFDL could be used. DFDL is not intended to provide quality information about data.

4 Conclusions

As mentioned in this trade-off, the DFDL specifies how bytes and bits should be interpreted or how low-level types are assembled into forms like arrays, etc. Therefore there is no direct way of describing or locating corrupted data using DFDL, like it can be done using the current version of SAFE.

Taking into account the information provided in the previous chapters, there are no clear advantages of replacing the current approach on SAFE with DFDL to specify corrupted elements.

Change Record

Issue	Revision	Date	Change Status	Origin
1	Pr1	17th October 2012	First Version	Miguel Vieira GMV

< End of Document >