ICALEPCS'09

International Conference on Accelerator and Large Experimental Physics Control Systems
October 12 – 16, 2009, Kobe International Conference Center, Kobe , Japan

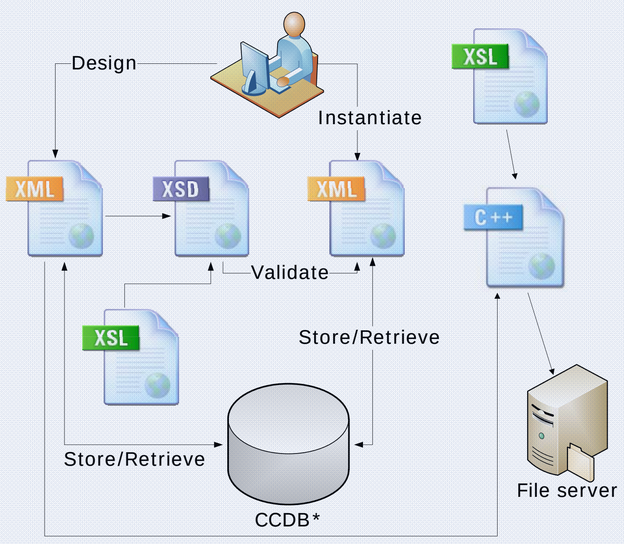
FESA 3.0: Overcoming the XML/RDBMS Impedance Mismatch

M. Peryt, M. Martín Márquez , CERN, Geneva, Switzerland

Abstract

The Front End System Architecture (FESA) framework developed at CERN takes an XML-centric approach to modelling accelerator equipment software. Among other techniques, XML Schema is used for abstract model validation, while XSLT drives the generation of code. At the same time all the information generated and used by the FESA framework is just a relatively small subset of a much wider realm of Controls Configuration data stored in a dedicated database and represented as a sophisticated relational model. Some data transformations occur in the XML universe, while others are handled by the database, depending on which technology is a better fit for the task at hand. This paper describes our approach to dealing with what we call the "XML/Relational impedance mismatch" – by analogy to Object/Relational impedance mismatch – that is how to best leverage the power of an RDBMS as a back-end for an XML-driven framework. We discuss which techniques work best for us, what to avoid, where the potential pitfalls lie. All this is based on several years of experience with a living system used to control the world's biggest accelerator complex.

FESA* WORKFLOW



FESA: Front End System Architecture
CCDB: Controls Configuration Database

DATA REPRESENTATIONS: XML VS RDBMS

```
<equipment-model xmlns:xsi="..." xsi:noNamespaceSchemaLocation="...">
    <information fwk-version="2.10" name="TestFullFesaClass" version="37"/>
    <standard-class>
        <interface>
            <device-interface>
                <std-setting-property name="Setting">
                    <simple>
                        <data-field-ref-item field-name-ref="multiplexedField"/>
                    </simple>
                </std-setting-property>
                <property name="ManageArray2D">
                    <complex>
                        <value-item name="custom2DArray">
                            <array2D type="short">
                                <dim1>10</dim1>
                                <dim2>10</dim2>
                            </array2D>
                        </value-item>
                    </complex>
                </property>
            </device-interface>
        </interface>
    </standard-class>
</equipment-model>
```

XML: hierarchical, nested, polymorphic.

RDBMS: flat, normalised, coherent.

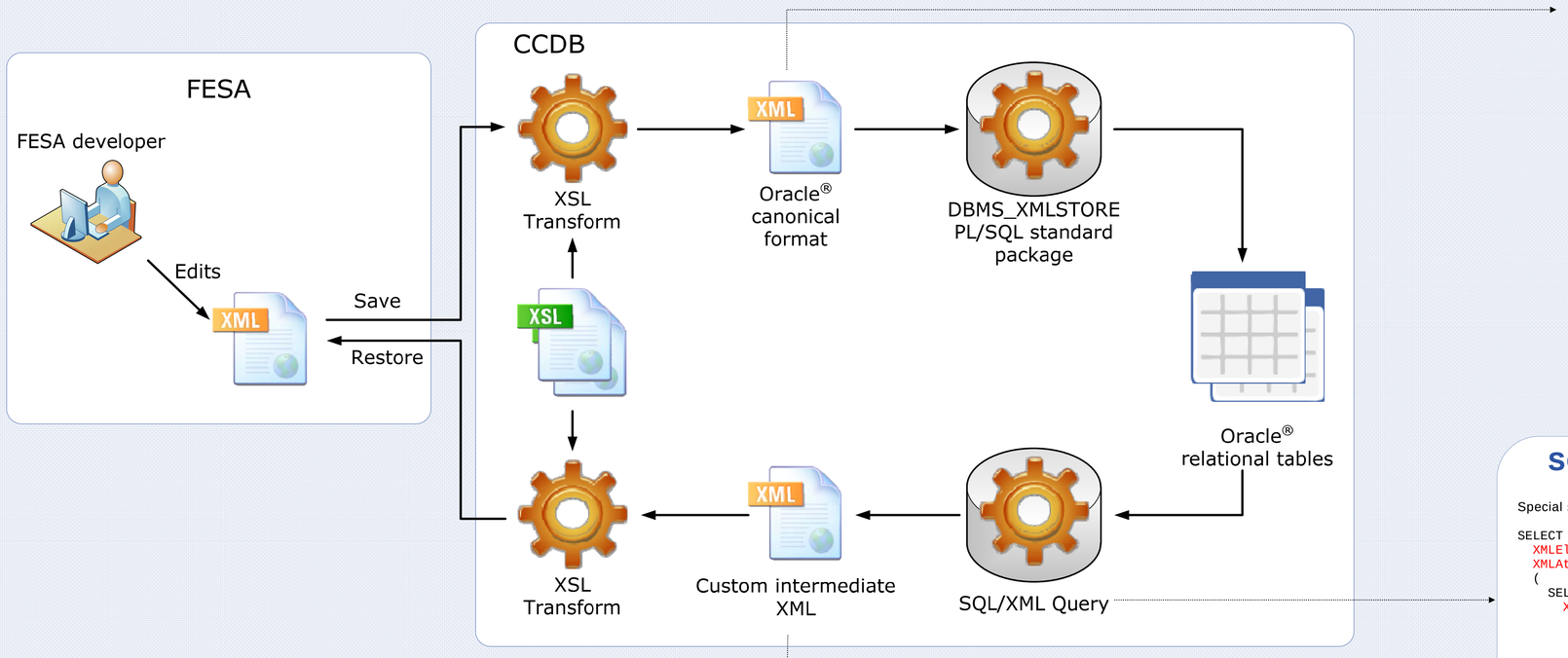
Bi-directional transformation required to preserve XML DOM fildelity (all information is kept except insignificant whitespace).

```
Table FESA_CLASSVERSIONS
CLASSNAME          VERSION CLASS_TYPE     FRAMEWORK_VERSION
------------------ ------- -------------- --------------------
TestFullFesaClass      37 standard-class 2.10


Table FESA_PROPERTY
CLASSNAME          VERSION PROPERTY_NAME        PROPERTY_TYPE IS_ALARM PROPERTY_ORDER IS_READ IS_WRIT PROPERTY_SUBTYPE          PROPERTY_SCOPE
------------------ ------- -------------------- ------------- -------- -------------- ------- ------- ------------------------ ------------------
TestFullFesaClass      37 Setting              SIMPLE        NO                    3 YES     YES     std-setting-property     device-interface
TestFullFesaClass      37 ManageArray2D        COMPLEX       NO                    9 YES     YES     property                 device-interface


Table FESA_IO_ITEM
CLASSNAME          VERSION PROPERTY_NAME        IO_ITEM_NAME         IO_ITEM_CLASS        FIELD_NAME       FESA_IO_TYPE DATA_TYPE   DIM1  DIM2
------------------ ------- -------------------- -------------------- -------------------- ---------------- ------------ ---------- ----- -----
TestFullFesaClass      37 Setting              multiplexedField     data-field-ref-item  multiplexedField
TestFullFesaClass      37 ManageArray2D        custom2DArray        value-item                            array2D      short         10    10
```

TECHNOLOGICAL CHOICES



ORACLE® CANONICAL FORMAT

The format that **maps 1:1 to database tables**:

```
<?xml version="1.0"?>
<ROWSET>
    <ROW>
        <CLASSNAME>TestFullFesaClass</CLASSNAME>
        <VERSION>37</VERSION>
        <PROPERTY_NAME>ManageArray2D</PROPERTY_NAME>
        <PROPERTY_TYPE>COMPLEX</PROPERTY_TYPE>
        <IS_ALARM>NO</IS_ALARM>
        <PROPERTY_ORDER>9</PROPERTY_ORDER>
        <IS_READABLE>YES</IS_READABLE>
        <IS_WRITABLE>YES</IS_WRITABLE>
        <PROPERTY_SUBTYPE>property</PROPERTY_SUBTYPE>
        <PROPERTY_SCOPE>device-interface</PROPERTY_SCOPE>
    </ROW>
    <ROW>
        <CLASSNAME>TestFullFesaClass</CLASSNAME>
        <VERSION>37</VERSION>
        <PROPERTY_NAME>Setting</PROPERTY_NAME>
        <PROPERTY_TYPE>SIMPLE</PROPERTY_TYPE>
        <IS_ALARM>NO</IS_ALARM>
        <PROPERTY_ORDER>3</PROPERTY_ORDER>
        <IS_READABLE>YES</IS_READABLE>
        <IS_WRITABLE>YES</IS_WRITABLE>
        <PROPERTY_SUBTYPE>std-setting-property</PROPERTY_SUBTYPE>
        <PROPERTY_SCOPE>device-interface</PROPERTY_SCOPE>
    </ROW>
</ROWSET>
```

SQL/XML QUERY AND ITS OUTPUT

Special set of **SQL functions**:

```
SELECT
  XMLElement("FESA_CLASSVERSION",
  XMLAttributes(t1.classname AS "CLASSNAME", t1.version AS "VERSION"),
  (
    SELECT
      XMLAgg (
        XMLElement("PROPERTY",
          XMLAttributes(t2.property_name as "NAME", t2.property_type as "TYPE")
        )
      )
    FROM FESA_PROPERTY t2
    WHERE t2.classname = t1.classname
      AND t2.version = t1.version
      AND (t2.property_name = 'Setting' or t2.property_name = 'ManageArray2D')
  ) AS "PROPERTY_LIST")
FROM FESA_CLASSVERSIONS t1
WHERE t1.CLASSNAME = 'TestFullFesaClass'
  AND t1.version = 37;
```

Generates **XML directly from database** tables:

```
<FESA_CLASSVERSION CLASSNAME="TestFullFesaClass" VERSION="37">
    <PROPERTY_LIST>
        <PROPERTY NAME="ManageArray2D" TYPE="COMPLEX"/>
        <PROPERTY NAME="Setting" TYPE="SIMPLE"/>
    </PROPERTY_LIST>
</FESA_CLASSVERSION>
```

CUSTOM INTERMEDIATE XML

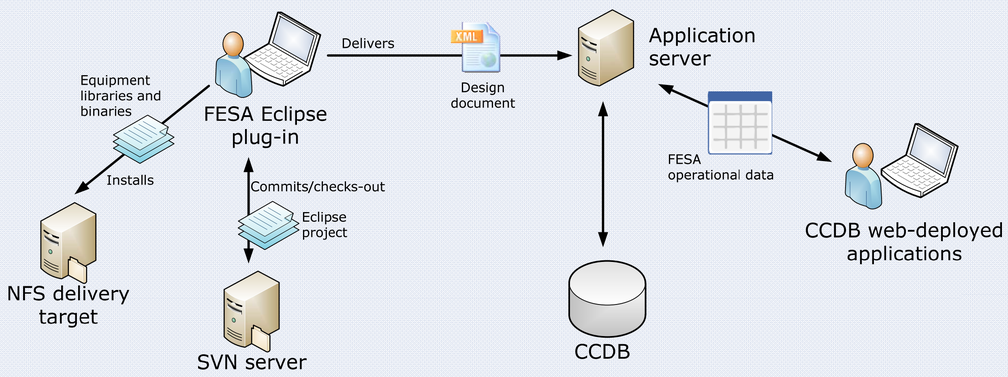
SQL/XML does not support dynamic XML element names, hence the need for **custom intermediate XML** with predefined element names and **dynamic data represented as attributes**:

```
<device-instance name="LEITestFullFesaClassDev1" timing_domain_name="LEI">
    <device-field name="customByte" persistency="FINAL">3</device-field>
    <device-field name="customTypeField" persistency="FINAL">VALUE1</device-field>
</device-instance>
```

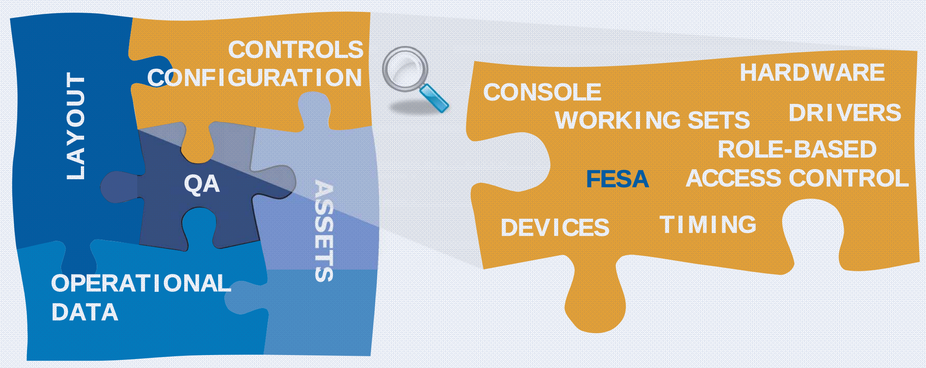
An XSL transformation then converts it to the representation expected by FESA – with **attribute values becoming element names**:

```
    <LEI-device-instance name="LEITestFullFesaClassDev1">
        <customByte FINAL="3"/>
        <customTypeField FINAL="VALUE1"/>
    </LEI-device-instance>
```

FESA 3.0 ARCHITECTURE



FEDERATED DATA MANAGEMENT



Conclusion

Although FESA 2.10 is a production system that is widely adopted at CERN and beyond (GSI in Darmstadt), we can consider the road that has been covered until and including this version a learning process. When the development started the XML technologies were still considered to be emerging and the subject of XML/Relational mapping was not very well explored. On the RDBMS side, the support for XML was also evolving at a rapid pace.
The FESA data management was rethought and re-implemented. The technological choices for FESA 3.0 XML Processor, biased towards Oracle® solutions, are fully in line with the strategy in place for the CERN accelerator controls. Inevitably, these implementations are difficult to export to other environments.

References

[1] M. Arruat et al., "Front-End Software Architecture", ICALEPCS'07, WOPA04.
[2] M. Arruat et al., "Use of XML Technologies for Data-Driven Accelerator Controls", ICALEPCS'05, PO2.094-5.
[3] R. Billen et al., "Accelerator Data Foundation: How It All Fits Together", ICALEPCS'09, TUB001 .
[4] D. Draper, "Mapping between XML and Relational Data", article at http://www.informit.com/ .