CMS NOTE 2001/057

The Compact Muon Solenoid Experiment

CMS Note

Mailing address: CMS CERN, CH-1211 GENEVA 23, Switzerland

November 2001

Detector Description Domain Architecture & Data Model

M. Case, Univ. of California Davis
M. Liendl, European Laboratory for Particle Physics (CERN) & Technical University Vienna
F. van Lingen, European Laboratory for Particle Physics (CERN) & Eindhoven Univ. of Technology

Abstract

This document discusses the detector description architecture for CMS. It describes the model and technology for storing data. Finally this document discusses other detector description efforts, and integration projects that use XML for storing, representing and exchanging data and compare it with the approach described in this document. This document is the basis for the design and implementation of the detector description software.

1 Introduction

Offline software such as simulation, reconstruction, analysis and visualisation are all in need of a detector description. Currently these applications use different independent sources for detector description data, which can lead to inconsistencies. Furthermore, several sources are sometimes needed to retrieve the right detector description data. The clients for the detector description repository have several common but also many specific requirements for the detector description in order to build up their internal representations.

To achieve this in a consistent and coherent manner a common source of information, the detector description database will be consulted [9]. This database is based on data from several different sources and has to deal with the semantic and structural heterogeneity of these sources. Data from these sources will not change frequently. However, several applications need to combine this detector description data with data that changes frequently, such as calibration data. Data will be accessed via a mediator that combines frequently changing data from calibration with data from the detector description database. Combining data involves complex transformations since calibration data relates to specific parts of the detector, while the detector description database stores a compact, ideal view of the detector. Different clients will have different views on the data, depending on their needs.

XML [16] has been chosen for several reasons:

- Most of the sources export XML.
- Several sources within this domain consist of flat files.
- XML applications based on standards such as Xpath [18] and Xquery [19] can be used for transformation and merging data.
- Tools are available that allow users to browse and edit XML data in a user friendly way.
- XML schema [17] allows for a flexible and type safe description of the data.
- XML parsers are available that validate the data, and can be used to import data in client applications.
- Data can be transformed to other XML formats.
- Due to the lack of manpower, one of the aims is to use standards that are supported by open source applications

Furthermore, numerous tools are being developed to support users in editing and developing XML. The use of XML within data integration of databases and semi-structured data, and XML views is discussed in [1],[8], [14]. Within the current architecture XML is used on two levels. XML will be used to describe the schema of the detector description database. Furthermore, detector description data will be stored in XML. XML and related formats has been used in several projects for data integration [3],[7],[13], [21].

This note is structured as follows: Section 2 gives an architectural overview. Section 3 discusses the underlying data model for the detector description. This model is independent from XML. Section 4 discusses numbering schemes. They can be seen as part of the data model, but are treated separately. XML is used to store data. The physical layout for this is discussed in section 5. The transient model is discussed in section 6. Section 7 discusses viewpoints on detector description data. Meta data is discussed in section 8. Dependencies between different components within the architecture are discussed in section 9. Section 10 discusses related work. Section 11 contains the conclusions.

The following definitions will be used throughout this paper:

- Data source. A source that contains data, that is accessible for other clients and users. This can be a flat file, database, hardware, software applications.
- Data base. A data source with data management functionality.
- User. A "real" person.
- Client. A software application that interacts with other software in order to exchange data.
- Transient model. The model (software) through which detector description data and other physics related data will be accessed by the clients. This is also referred to as a mediator.

2 Architectural overview

Several architectural decisions have been taken, based on user requirements [10]:

[Decision 1]: Data of the first prototype will be stored in XML. XML tools and applications can be used in the translation from data from native sources to the detector description source.

[Decision 2]: The detector description database will be the "original" source for detector description data. Data from other sources will be transformed into the format of the detector description database.

[Decision 3]: Calibration data and other physics data not described in the detector description will be accessed by means of a mediator that merges this data with data from the detector description

[Decision 4]: The interface of the transient model towards clients of the detector description consists of methods/functions specific for the detector description domain, instead of a generic query language as the basis for this interface.

Figure 1. Shows an overview of the architecture. Sources that contain the current detector description (TZ file, CAD database,….) will be wrapped in XML and this is transformed into XML instances conform the schema used by the XML database. The detector description transient model (the mediator) will use this data described by the XML instances to handle data requests from clients. Certain clients want to merge this data with calibration data or other physics data such as alignment data. The detector description transient model will provide mechanisms for this. The transient model will be based on the client side requirements, while the detector description database and calibration data will have their own models to efficiently store data.

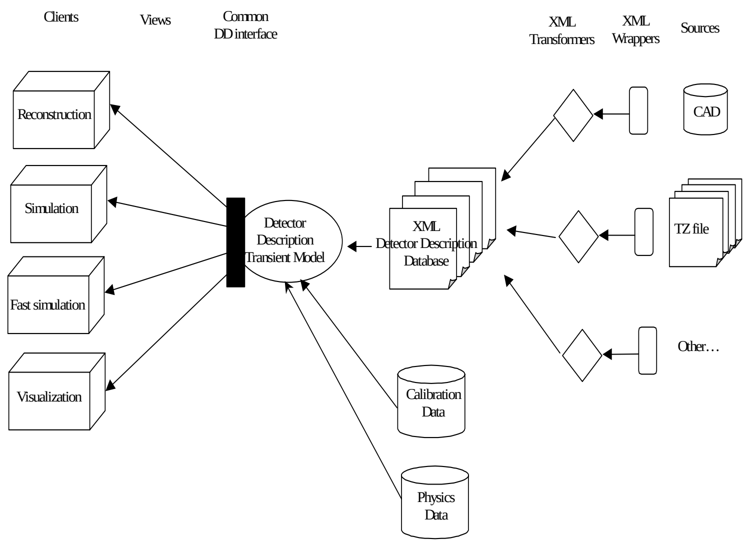


Figure 1. Detector Description Domain Architecture

An important requirement is the use of XML for storage of the detector description data. It is possible to store this data in a database and export XML to the users as an intermediate storage format (thereby still fulfilling the requirement that users can edit detector description data in XML). Within the detector description domain an ideal detector description(s) will be described. An ideal detector description is a simplified abstraction to describe the "real world" detector. Because of this abstraction data about the detector become more manageable. Within this note ideal detector description and detector description are used interchangeable. The current sources (TZ files, CAD database, …)for the detector description data do not change a lot. These sources will be the basis for the detector description database. The interface in the transient model towards the client can be used to construct different views on the detector description. In a general case such an interface could be a query language such as SQL, OQL [6], or XQL [19]. These are very general interfaces to data. The clients in the detector description domain require only a small subset of that. More important are parts of the interface that can not be expressed within these standard query languages. Furthermore, building a generic query language requires manpower, and a dedicated interface can be better optimised for the detector description domain.

3 Data model

The data model of the detector description is independent of the storage format (XML). This section will therefore not assume that XML will be used to store detector description data.

The detector is a machine. Data about machines can be represented as a bill of materials (BOM). A bill of materials is a tree in which the nodes represent certain detector parts, and the edges represent a "part of" relationship. Every node in the tree can be uniquely mapped to a "real" part or a collection of parts in the "real" detector.

The CMS detector description consists of a hierarchy of geometry descriptions. Furthermore, this hierarchy contains parameters that give a detailed description about elements within this description. This hierarchy can be compared with a bill of materials (BOM). However, a BOM is based on a tree structure. The detector description is based on direct acyclic multi graphs (Figure 2). Acyclic multi graphs can be unfolded into a tree. Within this paper a BOM refers to a direct acyclic multi graph structure

To prevent a "parts" explosion, the BOM is based on descriptions of parts. For example, the detector description will contain approximately 80,000 crystals. These 80,000 crystals can be described by five types of crystals. The BOM contains two layers of descriptions. A part type is described by a material, a solid, and a collection of positioned parts. A part is described by a part type and a position relative to a part type (parent part type). Furthermore, a part can have a parameterised position. This is a description of a part that places the part several times in a parent part type. Figure 2 shows an example of a detector description. The dotted lines point towards the parent part type of part. The number next to part B1 and part F1 denote a parameterisation of these parts. The detector will be described using the compact view. The compact view is used to construct the expanded view. The expanded view is a BOM. For example the part B1 of part type B is expanded to a part B12' and B11' in the BOM.

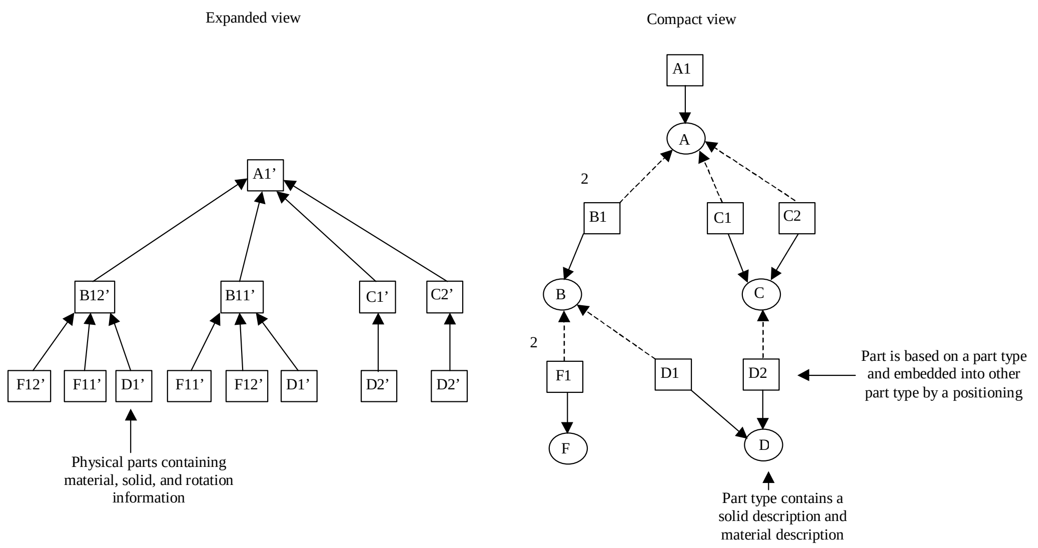

Expanded view
A1'
B12'
B11'
C1'
C2'
F12'
F11'
D1'
F11'
F12'
D1'
D2'
D2'
Physical parts containing material, solid, and rotation information
Compact view
A1
A
2
B1
C1
C2
B
C
2
F1
D1
D2
F
D
Part is based on a part type and embedded into other part type by a positioning
Part type contains a solid description and material description

Figure 2. Compact view that describes an expanded view (BOM)

Three types of positioning within the compact view:

- Parameter position. The position of n part types (of the same type) is described by a formula. For example there is a line of 10 long and a part has to be positioned 5 times along this line (part B1).
- Single position. A part type is only positioned once in a mother part type.
- Set of single positions. A user positions the same part several times in a mother part type, not using a parameter formula. Within the compact view in Figure 2 this is represented by C1 and C2.

Attached to every part type will be information about its material and geometry. Certain parts (or part types) have parameters specific for this particular part (type). For example, a crystal can have a parameter "reliability" while a silicon sensor can have a parameter "number of strips". Furthermore, different users will add their own specific parameters to certain detector parts or part types. Based on how detector data is currently used, four types of scopes will be offered in the detector description for attaching parameters to parts or part types.

- A parameter can be related to a part type
- A parameter can be related to the sub tree described by a part type.
- A parameter can be related to a (set of) specific part(s) within the expanded view.
- A parameter can be related to the sub tree of a (set of) specific part(s) within the expanded view.

Instead of parameter values, users want to be able to add constraints to a detector description. Certain constraints on data can be evaluated by XML validators. More complex constraints such as "The name of every tracker should start with TK" or "The value of luminosity for every crystal should be smaller than 5", are more difficult to validate. The scope of constraints is the same as discussed for parameters.

Specifications of detector parts can be related to each other. For example: the length of part A is three times the length of part B. A requirement is the one definition rule: Numbers should not be duplicated within the detector description. This implies that number within the detector description can be literal numbers of functions with references to other numbers. This leads to expressions such as:

LengthB=5, LengthA="3*LengthB"

Figure 3 shows an example of a detector description with constraints, parameters, rotation matrices and materials.

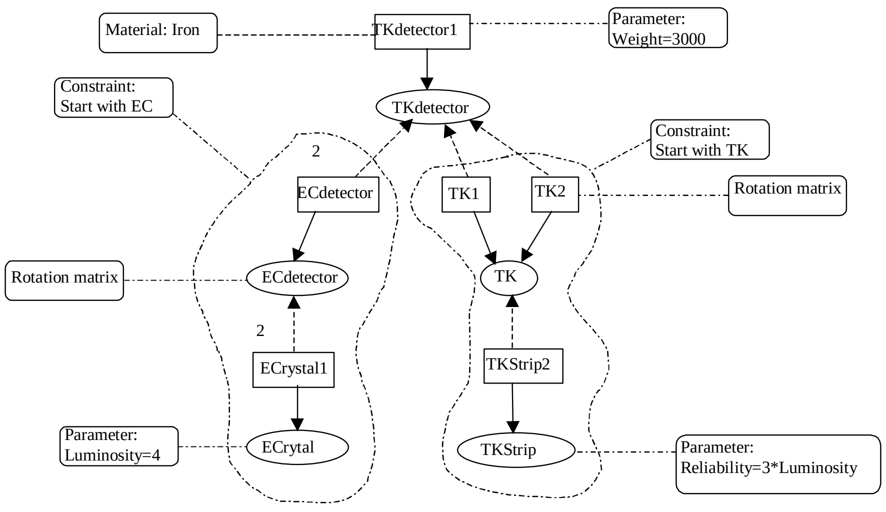


Figure 3. Compact view example

A complete BOM for the detector description would result in millions of parts. Therefore the detector description database will store an idealised representation of the "real" detector. A compact view reduces the size of the detector description data. For example there will be only five different types descriptions of crystals, instead of 80,000 descriptions. The compact view is based on the one definition rule stated in the user requirements. Data is not duplicated but referred to.

A constraint language, enables users to define constraints, which otherwise would have been "hard coded". A constraint language therefore results in a more transparent detector description. Constraints that can not be expressed in this constraint language have to be integrated in the transient model in a generic/extensible manner. An example of a generic constraint language is used in the Protégé project [13]. Xpath can be used as a simple mechanism to check constraints. Suppose "Part" tags have an attribute color. A constraint can be: A part has either color "red", "white", "blue" or "orange". This is equivalent with

```
Set_Of_Nodes ( //Part) = Set_Of_Nodes( //Part[@color="red"]) +  Set_Of_Nodes( //Part[@color="red"])
                       + Set_Of_Nodes( //Part[@color="red"]) +  Set_Of_Nodes( //Part[@color="red"])
```

In which Set_Of_Nodes represents the set selected by the Xpath, and + is the union operator.

The detector description database will be based on data from design and TZ files. The structure of these data sources contain a compact view.

4 Numbering schemes

The detector description database will store a direct a-cyclic graph that describes the structure of the detector (as discussed in the previous section) using "is a part of" relationship. The expanded view based on data from this graph needs to be related to other data outside the detector description database, for example calibration data. A piece of calibration data is related to one single detector part.

This is done via a numbering scheme. A numbering scheme is a unique numbering on the elements of the expanded view. This unique identification is used to correlate data from the detector description with data from other sources. However, the expanded view is not stored. Therefore this numbering scheme needs to be generated from the compact view.

The detector description software will generate a numbering scheme. But several clients have their own numbering scheme for various reasons. The main idea behind these different numbering schemes is to have efficient access to the data specific for a certain client. The concept of number scheme is to some extent misleading. If we look at it from a database point of view, the detector description numbering schemes can be compared with object id's. The other number schemes can be seen as indexes on data.

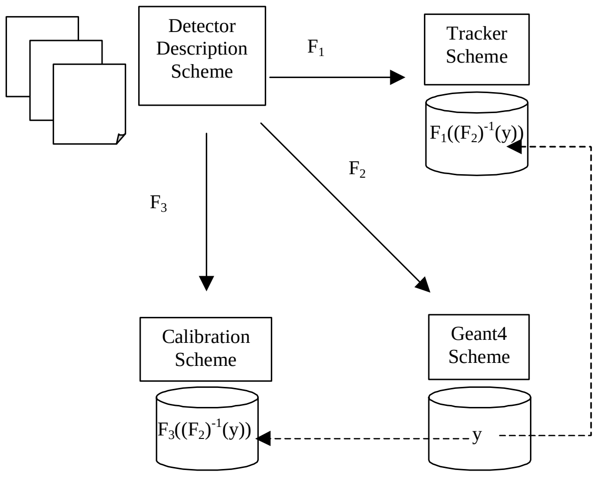


Figure 4. Correlating data from different sources.

Data generated by applications (clients) using different numbering schemes, have to be correlated, implying a need for mappings. Mappings from the detector description numbering scheme to any other numbering scheme will not always be surjective. Certain domains only apply a numbering scheme on certain parts in the detector description. Data generated from different clients can be correlated by using mapping functions and the inverses (if the latter exists). Figure 4 shows an example of data correlation. The next sections describes two examples of a numbering scheme.

Geant 4 will import the direct a-cyclic graph from the detector description domain and generates its own internal representation. Within the detector description certain parts (nodes in the a-cyclic graph) are "empty". They are place holders that are used to cluster parts together. Clustering is helpful in giving more structure to the data files. When generating the internal representation, Geant 4 will delete these nodes. Furthermore, the detector description domain can use certain shapes not known to Geant4. These shapes are related to a node in the a-cyclic graph. Geant 4 will decompose the nodes (parts of the detector description database) in parts that are used within Geant 4. For example if it is possible to use a "curve" shape in the DDD, Geant 4 could decompose this in a set of points where each point resembles a part (provided Geant 4 does not have any knowledge about curve shapes). This decomposition will be related to parameter nodes used within the detector description. Geant 4 will therefore change the direct a-cyclic graph from the DDD in two ways: delete nodes and decompose nodes. This will be done using the following restriction: A decomposed node will never be deleted. Figure 5 shows an example.

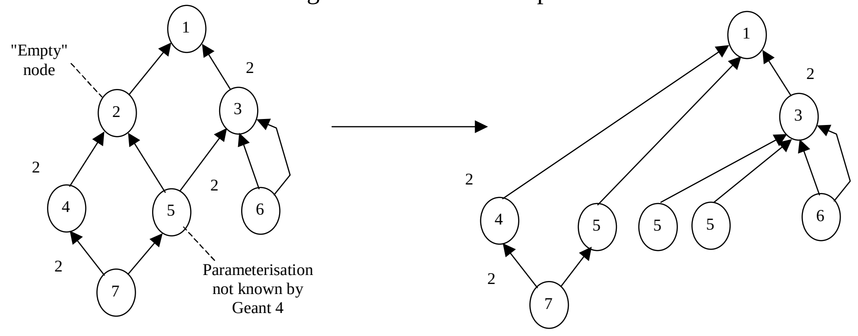


Figure 5. Transforming a direct a-cyclic graph to Geant 4 form

Another example is crystal numbering. Within the detector description numbering scheme, crystals receive an arbitrary number. This is based on position in the expanded view tree and type of crystal (there are 5 different types). However, within the "real" detector, crystals are positioned in a 2 dimensional manifold and certain applications want to number crystals starting from 1 to 80,000 without making a distinction between different crystal types. Figure shows and example of numbering of crystals (here there are two types of crystals).

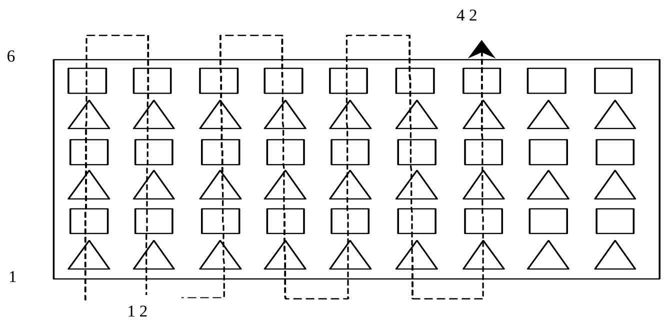
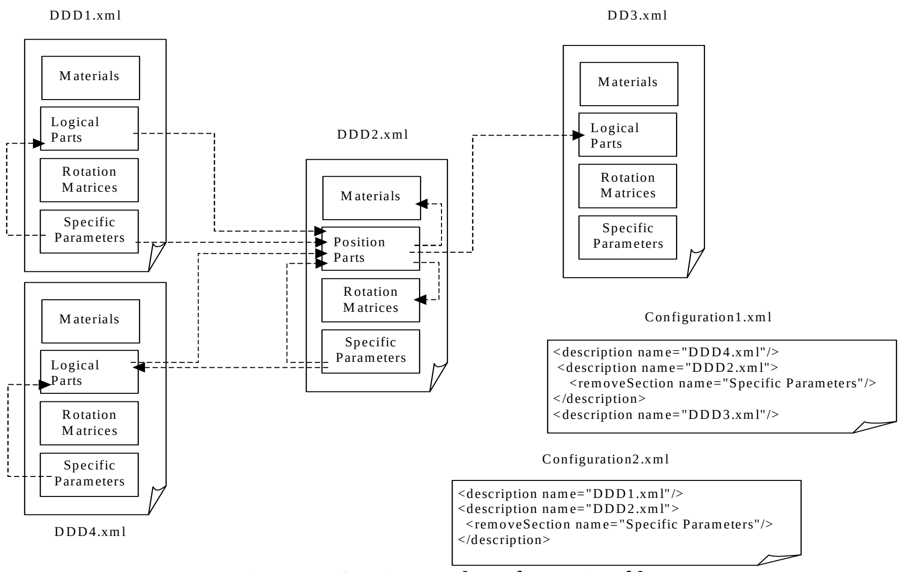


Figure 6. Crystal numbering scheme (with two types of crystals

5 Data storage

Section 3 discussed a structure for representing data based on acyclic multi graphs. The "one definition rule" requirement, and multiple versions of the detector description prevent a one to one mapping from the hierarchical structure of the detector description to the hierarchical structure of XML files. Furthermore, the amount of data will be to large to store in one file. Therefore there will be cross file data references.

Detector description files contain a number of sections: geometry, constraint, parameter, logical part, positioned part, rotation and material sections. Within these sections users define their description according to the detector description XML schema. Within the detector description schema, predefined references will be used to refer to material, logical parts, physical parts, geometry, and rotations. Next to the detector description files, are configuration files. The configuration files contain a list of files and section that constitute a detector description. The configuration files define a "version" of the detector. Different groups can have different configuration files.



Configuration1.xml
```
<description name="DDD4.xml"/>
 <description name="DDD2.xml">
   <removeSection name="Specific Parameters"/>
</description>
<description name="DDD3.xml"/>
```

Configuration2.xml
```
<description name="DDD1.xml"/>
<description name="DDD2.xml">
  <removeSection name="Specific Parameters"/>
</description>
```

Figure 7. Sections and configuration files

Figure 7 shows an example of four cross-referenced detector description files and two related configuration files.

Xlink and Xpointer [14] provide a generic reference system for XML data. However, this is a verbose notation. Within the detector description domain the types of references are known. This leads to a less verbose notation. Furthermore, human readability is a requirement for the detector description XML files. A generic notation used in Xlink does not enhance readability.

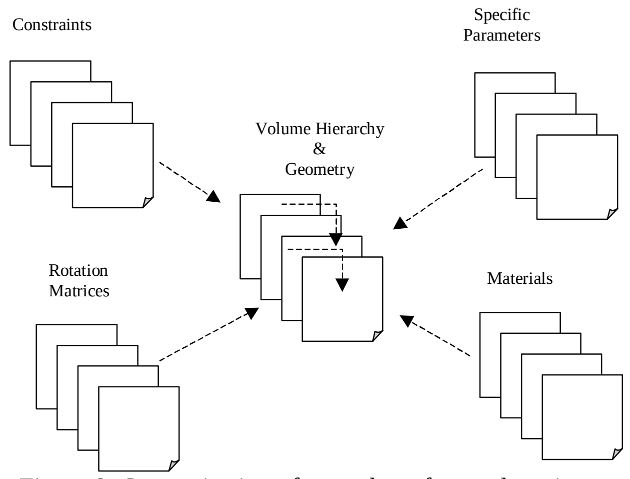


Figure 8. Categorisation of a number of named sections.

There are many possibilities to store detector description data. Another way to store detector description data is to "split" data into separate "collections" of files. This structure does not improve readability because data about a detector part is scattered in multiple files. The user cannot organize data in a convenient manner. Notice that the storage structure discussed previously enables to construct the structure of Figure 8 (one section per XML file) but does not enforce it.

6 Transient model

The transient model provides the interface to the detector description data and is able to create expanded views. It contains mechanisms to correlate detector description data with data from calibration. The transient model is related to the detector description domain, but the basis of the transient model will be the manipulation and combination of trees and direct a-cyclic graphs. This structure can be found not only in the detector description domain, but in many data sources dealing with detector data [11]. One of the design goals is that the core of the transient model can be re-used in another domain that also deals with trees and direct a-cyclic graphs. This implies that the transient model should be independent from the storage mechanism of the detector description that needs to deal with (XML). Other approaches do have a tight coupling between the storage mechanism, transient model and client applications [2],[4],[5]. These approaches focus on detector description. Within this note we emphasise the underlying structure of a detector description as graphs and trees. Exploiting such model yields more generic applicable software.

7 Client viewpoints

The transient model exports an interface to its clients. Different clients will use this interface in different ways, because of their internal representation of the detector. Thus creating different "viewpoints" on the detector description data. Currently it is foreseen that the number of viewpoints based on the transient model will not be large. This suggests that the software that will manage these viewpoints shall be straightforward, or not be needed at all.

8 Transformers and meta data

This section discusses several possibilities for transforming detector description data into other formats. If data can be transformed from the CMS detector description format (CMSDD), in a sufficient coherent manner to other formats, it will be possible to use applications that support this other format. This has not been a requirement, but if users want different views on data (by means of transformations), viewpoint management and meta data become important.

Figure 9 shows an example of several transformations: Protégé has a powerful constraint language. This could be used to formulate constraints that can not be expressed in the constraint language currently within the CMS detector description. ATLAS has developed several applications for its detector description that could be used if there is a transformation to AGDD format. The format described in Figure 7 is flexible and general. However certain users could need a different view of these files for various reasons. Such as a user only wants to see one file that contains only parts described by a DDL2 schema. Browsers provide a convenient interface to look at the data. Different transformers can translate XML to different HTML representations.

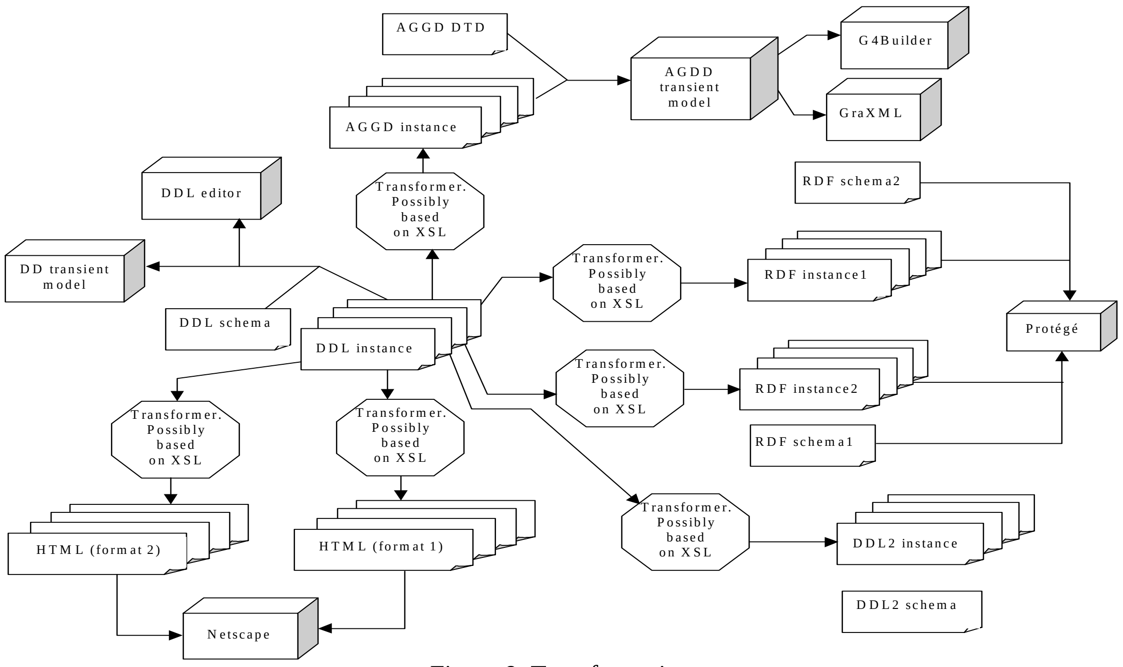


Figure 9. Transformations

An important requirement from the physics community is a DD format that is easy to understand and edit. This compromises the design of the schema: References that link data within different XML files are not based on Xlink/Xpointer. Furthermore, the configuration files "link" pieces of XML that form a "detector description" but are not based on Xlink. These standards provide very general but verbose linking and reference mechanisms.

Future work will concentrate on defining a schema that exploits these standards. This would allow for the use of more open source software for managing and manipulating XML data instead of developing this software within CMS. Physicists would edit/manipulate detector description data using the current schema definition. The data for this description is transformed from a description based on general XML concepts of describing data and relations between data (Figure 10).

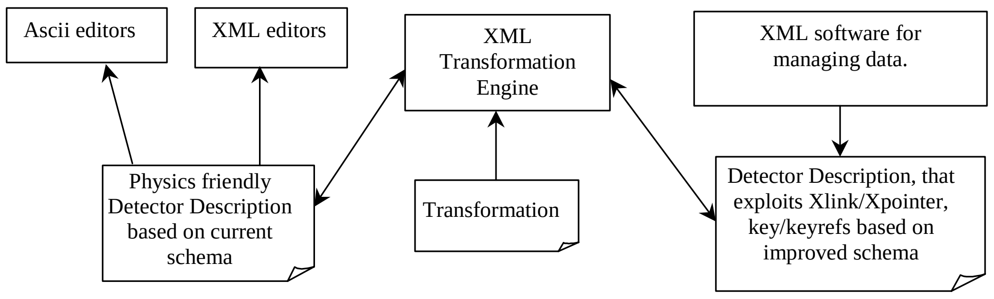

Ascii editors
XML editors
XML Transformation Engine
XML software for managing data.
Physics friendly Detector Description based on current schema
Transformation
Detector Description, that exploits Xlink/Xpointer, key/keyrefs based on improved schema

Figure 10. Transforming detector description data.

Creating different views on the detector description introduces the need for view management [1], [8]. What views are there, and what do they do? Within the first prototype of the DDD views will have a low priority.

It is important that users are able to get an overview of what transformations are available, where these transformations are located, and what group is responsible for this transformation. Furthermore, transformations, generates derived data sources. It is important to know what (derived) sources there are, from which sources it is derived, and when it was derived.

The following meta data will be attached to transformations:

- (Unique) transformation name
- Location
- Source format
- Target format
- Information loss. This is currently a parameter with one of the following values: None, Some, A lot
- Documentation. Describes the transformation and the loss of information in human understandable terms.

The following meta data is attached to data sources:

- (Unique) source name
- Location
- Source format
- Source type (derived, original, replicated)
- If derived, from which sources, and which transformations
- If replicated from which source
- If it is a derived or replicated source when was it last updated

The transformations and (derived) data sources are registered in a registration file. If a user requests an overview a script will consult the registration file and collect the meta data of the transformations and data sources. This meta data is combined into two graphs. A graph that displays the different transformations from an to different formats (Figure 9) , and a graph that displays the (derived) sources and dependencies.

9 Dependencies

The detector description project can be subdivided into several "self contained" packages. Figure 11 gives an overview of these packages. XML instances refers to the detector description data stored in XML files conform the DD XML schema. External Physics Data Sources refers to merging detector description data with data from these sources. The arrows indicate dependencies between packages. A thick arrow indicates a strong dependency. A thin arrow indicates a weak dependency. The initial goal is to have a prototype with limited functionality to show the proof of concept. Development will therefore focus on the packages that have a strong dependency: Wrappers, Transformers, XML Instances, XML schema, Transient Model, Client Views. Transformers I relate to the initial transformations that are needed to migrate detector description data to the new format. Transformers II relates to transformations to other formats such as RDF, HTML, and AGDD. Transformers can be specified in XSL or can be compiled source code.

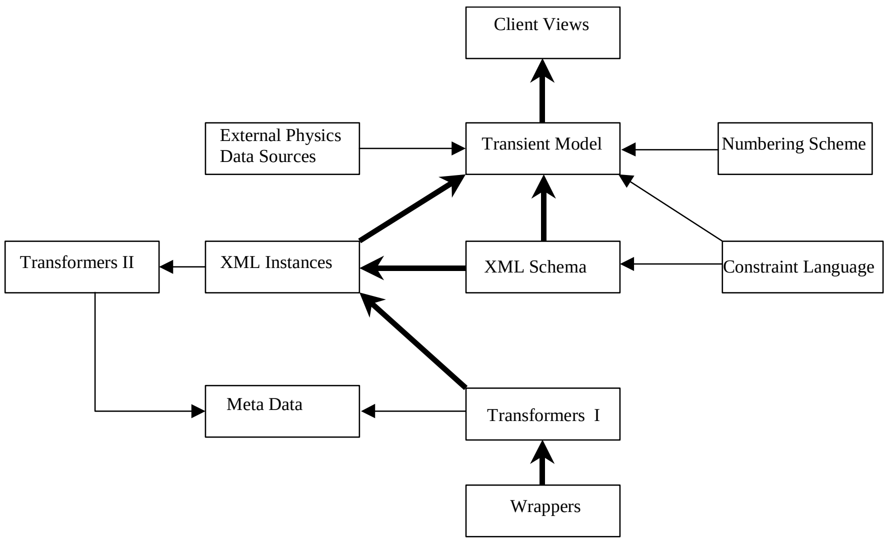


Figure 11. Dependencies between packages.

10 Related work

Several groups are developing detector descriptions [2],[4],[5]. These approaches have in common the use of XML. However, the approach described in this paper differs in several aspects (not all aspects are covered by other detector description groups):

1. De-coupling of the storage medium. It does not consider XML as "the" mechanism to store detector data. Within its architecture it will cater for "a" storage mechanism.
2. The focus is on the underlying model of the detector description. Trees, a-cyclic multi graphs and mappings between these structures. Since this structure can be found in many CMS sources, the software (transient model) can be reused within different domains.
3. The introduction of a constraint language. Not everything can be modelled. A simple constraint language is introduced to describe constraints. Since the constraint language is not rich enough (due to manpower constraints), the transient model contains a module to describe more complex constraints
4. Different sources will supply data for the detector description database. This data needs to be transformed.
5. Introduction of different numbering schemes for different clients (indexes).
6. Using references to numbers to comply with the one definition rule (e.g. x=3*y)
7. Possibility of transforming detector description data into other formats such as HTML. These transformations could be based on another XML standard: XSL
8. Relate this approach to well established data integration/base techniques such as indexing, databases, mediators, and meta data
9. Using XML schema, instead of DTD.

The following table shows a comparison of the different approaches:

Characteristics	CMS	ATLAS [2]	GLAST [4]	GDML [5]
1	Yes	Yes	Yes	Yes
2	Yes	No	No	No
3	Yes	No	No	No
4	Yes	Yes	Yes	Yes
5	Yes	No	No	No
6	Yes	No	?	Yes
7	Yes	*	*	*
8	Yes	No	No	No

9	Yes	No	No	Yes

Table 1. Overview of differences.

*possible, but not done

Numerous projects dealt with generic data integration, several of which were based on XML and mediators ([3],[7],[12]). Most of these projects deal with the general problem of integration. The focus is on finding high level specification mechanism for constructing mediators and sufficiently rich query mechanisms for retrieving data. Within the detector description domain and other domains such as construction, design, calibration, the models are related to trees and direct a-cyclic graphs. Therefore it is not necessary to use a very generic approach as described in other integration projects. Instead it is sufficient to focus on exploiting the underlying structure of a-cyclic graphs and trees as described in this note. With respect to integration projects such as [3],[7],[12] the mediator within the detector description domain is not as generic. However, it covers several areas for integration within CMS domain, of which detector description is a sub domain. Consequently this architecture could also be deployed within the other physics experiments ATLAS, LHCb and ALICE. These experiments have in common the use of tree structures and a-cyclic graphs.

11 Conclusions

From a database point of view, the approach discussed in this note deals with data integration on two levels. First data is integrated via a database using XML. Second a transient model (a mediator in database terms) merges data from this database with other sources. The mediator model is abstracted from the detector domain in terms of tree structures and acyclic multi graphs, and could be used within other integration efforts within CMS dealing with bill of materials. Creating different transformations to other domains (HTML, AGDD, RDF) might lead to requirements for managing these transformations. This could be supported by the introduction of meta data that describe the transformations, the sources and the relations between transformations and sources. Currently meta data has a low priority.

References

[1] S. Abiteboul, “*Views and XML*”, Proceedings. ACM Symp. on Principles of Database Systems, pp. 1-9, 1999.

[2] Christian Arnault, Stan Bentvelsen, Steven Goldfarb, Marc Virchaux, Christopher Lester, *“A generic approach to the detector description in Atlas”*, CHEP 2000

[3] C. Baru, A. Gupta, B. Ludascher, R. Marciano, Y. Papakonstantinou, P. Velikhov, V. Chu, *“XML-Based Information Mediation with MIX”* exhibition program, ACM Conf.on Management of Data SIGMOD99, 1999

[4] J. Bogart, D. Favretto, R. Giannitrapani, "*XML for detector description at GLAST"* Chep 2001

[5] Radovan Chytracek, "*The Geometry Description Markup Language"* Chep 2001

[6] Won Kim, "*Modern Database Systems"* Addison Wesley 1995

[7] H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and Jennifer Widom. *"Integrating and Accessing Heterogeneous Information Sources in TSIMMIS*". In Proceedings of the AAAI Symposium on Information Gathering, pp. 61-64, Stanford, California, March 1995

[8] Alon Levy, “*More on Data Management for XML”*, 1999 http://www.cs.washington.edu/homes/alon/widom-response.html

[9] M. Liendl, F. van Lingen, M. Case, T. Todorov, P. Arce, A. Furtjens, V. Innocente, A. de Roeck, "*The Role of XML in the CMS Detector Description"*, in proceedings of CHEP 2001

[10] M. Liendl, F. van Lingen, M. Case, "*Detector Description Requirments"*, CMS note in preperation.

[11] Frank van Lingen, "*Generic Data Integration Based on XML within a High Energy Physics Environment"* Mphil thesis, september 2001, University of the West of England

[12] Manolescu, I. Florescu, D. Kossman, D. Xhumari, F. Olteanu, D. *”Agora: Living with XML and relational”*, VLDB 2000 pp 623-626.

[13] N. F. Noy, R. W. Fergerson, & M. A. Musen. The knowledge model of Protege-2000: Combining interoperability and flexibility.2th International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000), Juan-les-Pins, France, . 2000.

[14] Jenifer Widom,"*Data Management for XML"*, IEEE Data Engineering Bulletin, Special Issue on XML, 22(3) 44-52, september 1999
[15] Xlink, http://www.w3.org/XML/Linking
[16] XML, http://www.w3.org/XML
[17] XMLSchema, http://www.w3.org/XML/Schema
[18] Xpath, http://www.w3.org/TR/xpath
[19] Xquery, http://www.w3.org/XML/Query
[20] XSL, http://www.w3.org/Style/XSL/
[21] Zachary G. Ives, Alon Y. Levy, Daniel S. Weld, Daniela Florescu, Marc Friedman. " *Adaptive Query Processing for Internet Applications*." IEEE Data Engineering Bulletin, Vol. 23 No. 2, June 2000.