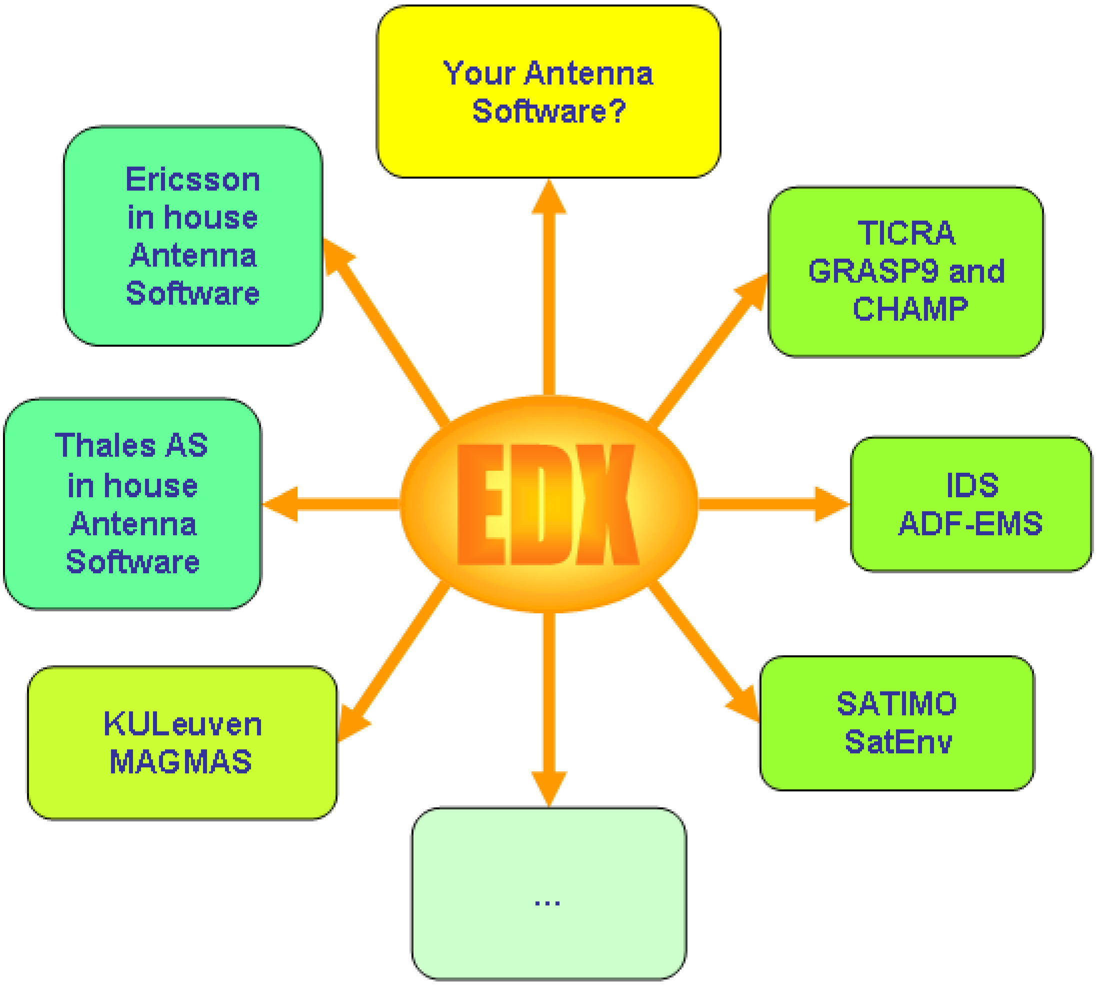
Electromagnetic Data Exchange Language

What is it?



April 2008

Marco Sabbadini and Poul Erik Frandsen

The increasing use of electromagnetic modelling in antenna design has led to the need for a common way to exchange data between the various software tools available in this field. ESA's efforts to promote such a standard reference are now producing results.

Over the past ten years, numerous developments have increased the reach of analytical techniques in antenna design, bridged the gap between them and purely numerical methods, produced a wealth of hybrid approaches, and improved the accuracy and performance of modelling of complex antenna systems in their operational environment. However, the very diverse nature of electromagnetic modelling techniques and the quest for the best modelling solutions make it necessary to use multiple tools in combination. This is necessary both to take advantage of the most efficient method for each of the various parts of an antenna system and to satisfy the varying requirements at different stages of the design and development cycle.

Most antenna modelling tools use proprietary data formats, while in most cases they handle the same physical entities. Transforming data from one format to another or writing software functions to do so is a costly and error-prone chore. The need for common way to describe the physical objects and quantities involved in the electromagnetic modelling of antennas is well recognised. A common data format, if accepted by the community at large, will lead to much easier interchange of data, with the potential of making a much wider range of modelling options available to all members of the antenna community.

Actions to fulfil this need have been promoted by ESA since the early 1990s. More recently the Multipurpose Antenna Design Simulator (MADS) project, funded under the European Union's fifth framework programme for research, technological development and demonstration activities (EU-FP5), has produced a first common format. The Electromagnetic Data Format (EDF) is based on NetCDF, a platform independent high-performance data storage format. Unfortunately, the limited resources available resulted in limited flexibility and the absence of a plain-text data file option.

To overcome these limitations, the Antenna Software Initiative Working Group of the Antenna Centre of Excellence (an EU-FP6 Network of Excellence) and the European Antenna Modelling Library team – working under ESA contract – have joined forces to develop the common Electromagnetic Data Exchange language (EDX).

A neutral, quite general and simple grammar, the Electromagnetic Markup Language (EML), has been developed to describe the actual content of data files. A set of agreed (but open to evolution) Electromagnetic Data Dictionaries (EDDs) establish the lexicon of the exchange language – that is, how to convey actual meaning using the EML grammar. Finally, a software library, the Electromagnetic Data Interface (EDI), allows standardised access to data written in this new language from C++, Fortran and Matlab® programs. The combination of these three elements is the Electromagnetic Data Exchange (EDX).

Handle the most common data sets
Offer an easy to understand human-readable format
Support the different needs of a wide variety of usages
Handle multiple representations of the same quantity
Store meta-data as well as special private data
Open to revisions and extensions
Robustness with respect to changes
Open and freely available library for access to data
Handle large amounts of data via high performance standards
Platform independence (both software and hardware)

The golden rules of the EDX development

When approaching the development of a data exchange language, three key questions must be considered:

- What data shall be handled?
- How shall the data appear in the file?
- How shall software tools access these files?

Since the EDX language should answer these questions on behalf of a rather large community of scientists and engineers, it was felt necessary to lay down a small set of basic requirements that could be shared by many.

Many robust data exchange solutions compatible with these requirements are available and selecting the right starting point already requires a significant ingenuity. Past experience within the antenna community has shown that a simple format based on a line-by-line description of the file contents would be severely limited. At the same time, very flexible formats having programming-language-like data constructors have proven to be rather difficult to use for the average developer, while high-performance solutions pose some accessibility problems for non-expert users. The need for human-readable files combined with that of handling large amount of data lead to the choice of XML, the leading reference for text-based data exchange.

Most often the data to be exchanged, like a tabulated antenna pattern, are a series of numbers with no specific meaning on their own. Their meaning needs to be conveyed separately, either using a fixed format specified by some document or by attaching this information to the data. Since this information is the most relevant for the human reader, EDX goes one step beyond. It starts by specifying the meaning and structure of data and then gives the data. A neutral, quite general and simple grammar, easily expressed in XML, has been developed to describe the content of data files, the Electromagnetic Mark-up Language (EML). It features the basic elements required to represent generic electromagnetic quantities, like the ability to specify scalar, vector and matrix quantities and to associate measurement units to them.

EML only provides the tools to specify how the information is conveyed, e.g. how different quantities are called or and structured, but it does not specify names and structures. A set of Electromagnetic Data Dictionaries (EDD's) establishes the lexicon of the exchange language, i.e. how to convey the actual meaning using the EML grammar. For example, the possible types of component projections and sampling grids for a far field are uniquely defined to make information exchange about antenna patterns possible.

Finally, a software library, the Electromagnetic Data Interface (EDI), allows standardised access to data in the EDX language. Of course, having the EML and the EDD's in place would be sufficient to guarantee successful data exchange. Still everybody would have to write functions to read and write the EDX files, with an enormous duplication of effort, simple mistakes making the actual exchange difficult and, last but not least, an open door to "slightly twist" the language to fit some specific purpose. A software library relieves developers from the burden to write their own access functions, avoids mistakes and provides a common baseline to which any other implementation can be compared for compliance with the EDX reference.

The structure of EDX can then be summarised using the following symbolic formula:

EDX = EDD's + EML+ EDI

Furthermore a set of utilities, the EDX Companion Tools, is under development to help the use and extension of EDX.

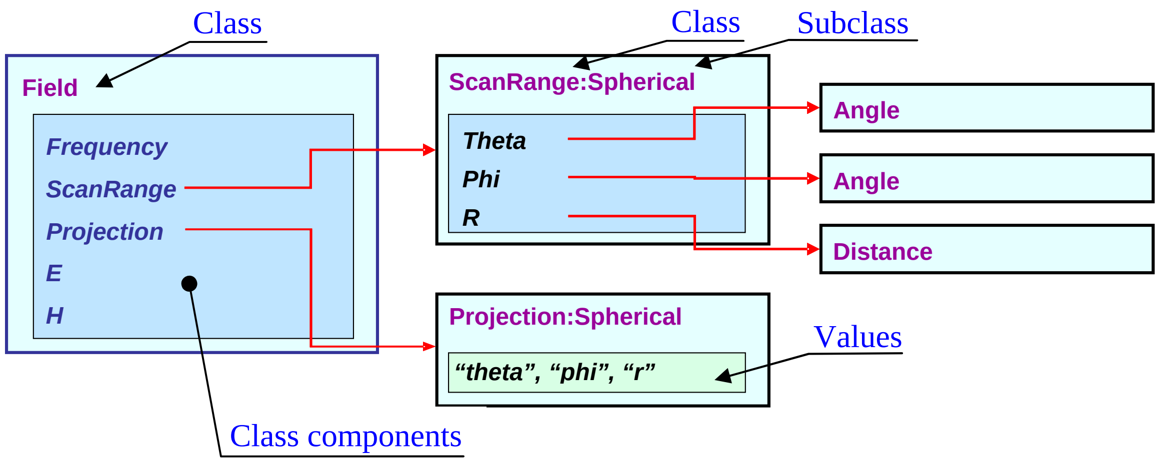
The meaning of data – the Data Dictionaries

In a Data Exchange Standard the Data Dictionary defines the meaning of data and the conventions for their exchange. The data dictionary defines *exactly and in detail* all the elements that shall appear in a data set.

Note, that the term "data set" is used – not "files". A data set is a collection of related simple data elements which as a whole models a physical entity such as a field, a bicycle tire or any other physical item. For the time being it is assumed that a file contains the data set, but it could as well be transmitted over a channel or it reside in a memory area.

While EML defines the grammar, a data dictionary contains and defines the 'words' to be used to describe the main physical entity, be that a field or a bicycle tire. In other words, it establishes the precise meaning of the data found in the data set, how they are represented numerically and the conventions adopted for their exchange; For instance, the implied time dependence in spectral representations (e.g. $e^{+j\omega t}$).

A physical entity is seldom atomic. Only scalar values are, all other entities have an intrinsic structure. Therefore each data set contains a number of elements, each one having its own definition and representation. Often these lower level entities appear in different data sets. The most obvious example is frequency; another is a reference system. It is worth noting that in most cases these lower level entities have a more abstract nature that he main one.



The basic structure of a physical quantity

Six data sets have been initially identified, namely:

1. Fields (near, far and spherical wave expansion)
2. Induced currents on various geometries
3. Green's function for layered structures
4. Circuit parameters – [S], [Y] and [Z]
5. Modal expansions
6. Geometry.

Currently the lexicon required for fields (near, far and spherical wave expansion) and for currents and meshes has been fully defined, while a draft exists for the global dictionary involving all the six data sets listed above.

The form of data – the EML

The grammar selected to describe their structure is a specialisation of XML inspired to NetCDF. The key idea is that a single data entity, the variable, is sufficient to the static representation of data stored into files. A variable may or not depend on other ones, via its domain, thus allowing the representation of sampled functions of n variables. A variable may have multiple components, each one of them being multi-dimensional. For example, an electromagnetic field sampled on a plane at multiple frequencies can be represented as:

```
Variable emField
 Domain frequency
 Domain samplingX
 Domain samplingY
 Component E (complex, dimension=3, unit=V/m)
 Component H (complex, dimension=3, unit=A/m)
end
```

A variable may also have no domains and its components may just be references to other variables, allowing the creation of hierarchies of aggregates. For example:

```
Variable aDouble ReflectorAntenna
 Component feedArray
 Component subReflector
 Component mainReflector
end
```

These examples highlight an important feature of EDX, if an extra domain or component is added to a variable, a tool not knowing about it is still able to use the rest of the variable data, just ignoring the extra information. In the same way it is possible to add extra variables to a data set without compromising its usability from unaware tools. This is a fundamental improvement with respect to usual situation.

EML files are organized in four main sections: Header, Declarations, Data and Application Data. The Header section contains general data like author, date, EDI version, etc. The Declarations section contains the definition of all variables as well as the values for small-sized ones. Larger amount of numerical data are found in the Data section, arranged in blocks, one for each of variable. Finally, the Application Data section is available for tools that need to save private data.

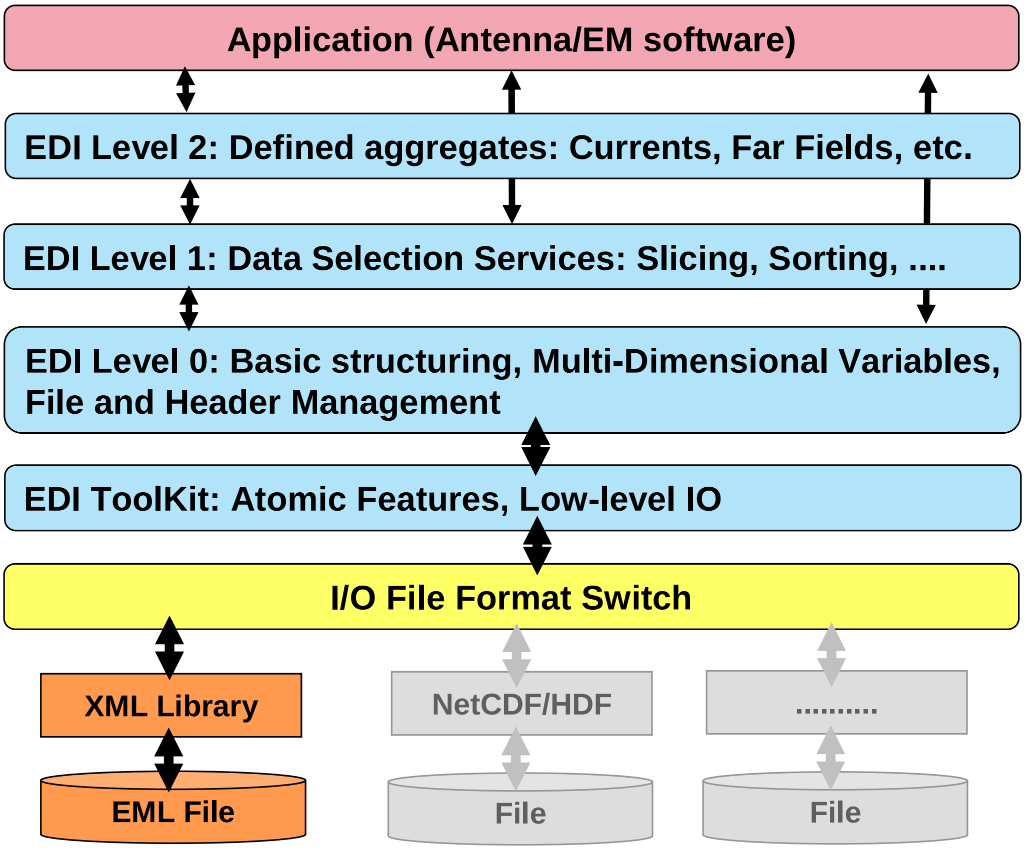
The handling of data – the EDI

The third element of EDX is a software library providing all functions required to access data. The Electromagnetic Data Interface is a relatively small library (a few thousands lines) written in C++ and equipped with application programming interfaces in C++, FORTRAN90 and MATLAB® (only Level 2 at the moment).

The library has a layered structure with each layer offering more advanced features compared to the lower one.

The main purpose of EDI is to simplify writing computer programmes using EDX, the highest level offers a single call access to complete data sets (although with very limited functionality) and more and more detailed access is available working at lower levels. The EDI Toolkit is a foundation layer and it is not accessible through the application programming interface.

As clearly shown in the figure below all functions offered by the library are independent from the actual format in which the data file is written, so that different ones can be used with no changes in the modelling tool.



The overall structure of the Electromagnetic Data Interface (EDI)

EDX Companion Tools

On top of the tree core elements described so far, it is the intention to offer a number of complementary utilities to enhance the use of EDX. These utilities are being developed and include:

1. EDX Tool: an I/O function generator
2. A generic EDX data browser
3. A generic EDX data visualisation tool
4. A validation tool for EDX implementations.

Furthermore the possibility to develop a CAD data import filter associated to a Geometry and Materials Data Dictionary is being explored.

The EDX tool is a code generator that takes as input a formal definition of a data dictionary, written in a language very close to EML, and a set of programming language fragments corresponding to the different constructs of the language and produces I/O functions for the data dictionary quantities by assembling and completing the fragments.

The Generic EDX data browser allows the navigation through data stored in the EDX language. The data organisation and values are visualised in text form in a way that makes them easy to understand for human users with a good knowledge of antenna engineering and possibly no knowledge of computer science and software engineering.

The Generic EDX data visualisation tool displays and prints the data stored in the EDX language. The data are visualised in graphic form in a way that makes them easy to understand for antenna engineers and following the common practices of this discipline.

The Validation tool for EDX implementations performs the validation of I/O functions providing access to data stored in the EDX language, providing a detailed report of all non-compliances found in the files generated by them. It is accompanied by a set of reference data files in the EDX formats to check input capabilities of other tools.

The CAD data import filter is intended to be a simple tool able to translate geometrical descriptions found in CAD files and directly compatible with a Geometry and Materials Data Dictionary into an EML file. Some advanced capabilities, like geometry healing and a smart visualisation of 3D shapes may also be included.