Extensible earth observation data catalogues with multiple interfaces

Patrick Harms [(1)], Stephan Kiemle [(2)], Daniele Dietrich [(2)]

*[(1)] **Werum Software and Systems AG***
Wulf-Werum-Straße 3, D-21335 Lüneburg, Germany
EMail: Patrick.Harms@werum.de

*[(2)] **German Aerospace Centre (DLR)***
German Remote Sensing Data Centre (DFD)
Oberpfaffenhofen, D-82234 Weßling, Germany
EMail: Stephan.Kiemle@dlr.de, Daniele.Dietrich@dlr.de

ABSTRACT

Earth observation data products must be made available for a growing number of user groups. In general access to this data is provided through catalogues enabling searching for products, retrieving their metadata, providing order options and ordering them. One challenge of designing catalogues is to address the very diverse demands of a broad variety of users. Beneath referring to the graphical user interface, those demands include at a major extend the need of integrating or combining the functionalities of specific catalogues into external systems, requiring catalogues to provide accessibility through a variety of protocols like CIP, SOAP, OpenGIS, etc. Further catalogues must be extensible to support future missions and therefore new types of products including new metadata structures. At least the functionalities for ordering products must be generic to be adaptable for future requirements as well.

This paper gives a more detailed overview about the challenges going along with designing catalogue implementations and describes a generic structure applicable for new catalogues considering the earth observation data catalogue EOWEB® (Earth Observation on the WEB) provided by the DLR as a reference implementation. The focus will be the modelling of metadata structures and the generation of protocol specific functionality for accessing the metadata of products directly without a protocol translation. Further this paper will show up possibilities for providing protocol independent order option retrieval mechanisms and ordering facilities. At least there will be a short discussion concerning the technologies currently used in EOWEB® and possible substitutions.

Keywords: Catalogue, Metadata, Order Options, Ordering, CIP, OpenGIS

INTRODUCTION

Earth observation data products must be made available for a growing number of user groups. In general access to this data is provided through catalogues enabling searching for products, retrieving their metadata, providing order options and ordering them. One challenge of designing catalogues is to address a broad variety of users in different roles.

Different user groups make different demands on such catalogues. At the minimum these refer to the graphical user interface concerning query definition, metadata representation and their depth of details as well as order definition, submission and monitoring. But due to the growing need of integrating or combining the functionalities of specific catalogues into external systems, the requirements also refer to non graphical interfaces. I.e. a catalogue must provide accessibility through a variety of protocols like CIP, SOAP, OpenGIS, etc. Those interfaces can then be used by different, user group specific implementations to create graphical views on a catalogue.

Concerning the registration and representation of products provided, a catalogue must be extensible to support future missions and therefore new types of products including new metadata structures. Further the functionalities for ordering products must be generic to be adaptable for future requirements as well.

This paper gives a more detailed overview about the challenges going along with designing catalogue implementations and describes a generic structure applicable for new catalogues considering the earth observation data catalogue EOWEB® (Earth Observation on the WEB) provided by the DLR as a reference implementation. The focus will be modelling of metadata structures and - based on the metadata model - generating protocol specific functionality for accessing the metadata of products directly without a protocol translation. Further this paper will show up possibilities for providing protocol independent order option retrieval mechanisms and ordering facilities. At least there will be a short discussion concerning the technologies currently used in EOWEB® and possible substitution.

CATALOGUE INTERFACES

Interfaces to earth observation data catalogues mostly follow similar principles. The provided functionality usually allows collection discovery, product discovery based on conditions, product metadata and quick look retrieval as well as – if supported by the interface – order option retrieval and ordering. The differences between the protocols are mainly located in terms like

- connection oriented or connection less
- authentication and authorization
- collection discovery mechanism
- product discovery mechanism and condition specification
- data structures of product metadata
- order option retrieval mechanisms
- order option data structures
- order submission mechanisms
- order data structures

If a catalogue needs to provide several different protocols being divergent in those points, this usually leads to an architecture where not only one catalogue but several catalogues are set up, each supporting another protocol. Further each of the catalogues has its own database, because the data structures have different approaches.

The major differences of the protocols are to be found in the mechanisms they use for connecting clients to servers or executing requests. Anyway the data structures used within the requests to transfer product metadata, order options or orders are very similar. The reason for this is that the data to be transferred carries almost the same content. As an example, each earth observation data product usually has a spatial coverage. This could be given as a polygon, a box or a single point. To be able to transport this information a protocol must define appropriate data structures. These definitions usually result in an abstract syntax like:

- the spatial coverage is either a polygon, a box or a point
- a polygon is an ordered set of points (at least three)
- a box is described by a northern latitude, a western longitude, a southern latitude and an eastern longitude
- a point is a pair of a longitude and a latitude

Although this example does not cover all known shapes representing a spatial coverage, it shows the equality between different data structures, which are defined for different protocols, but which are in general being able to carry the same information.

Many protocols define a basic structure for such metadata. This structure only slightly differs from protocol to protocol. Some protocols even refer to other protocols or standards to take over identical or similar structures. Further the protocols usually offer an opportunity to extend their basic metadata structure to allow the usage of the protocol also for product metadata which is more complex than the basic structure itself.

All these statements also apply for data structures representing order options or complete orders.

The next section will now give a more detailed look on data structures in general and how they might differ although carrying the same information.

Data Structure Diversity

Data structures representing product metadata, order options or orders can be seen as trees. A tree consists of nodes where each node might have children. There is at least one node without a parent node being the root of the tree and there are one or more nodes having no children and therefore representing the leafs of a tree.

If two different data structures represent the same data set (product, order options, order), they can be seen as two different trees. There are three cases for any information or semantic meaning carried by any node of these trees:

- the semantic meaning or information of node x of tree 1 is the same as of node y of tree 2
- there is no semantic or informational equivalent node y in tree 2 for node x of tree 1
- there is no semantic or informational equivalent node x in tree 1 for node y of tree 2

These cases are shown in Figure 1. This figure also shows cases where information or semantic meaning stored in both trees is located at two completely different positions in the trees. The figure does not consider a case, where a non-leaf node in one of the trees represents a leaf node in the other tree, as this should not be the case if real information is only carried by terminal nodes and non-terminal nodes are only used for structuring the terminal nodes.

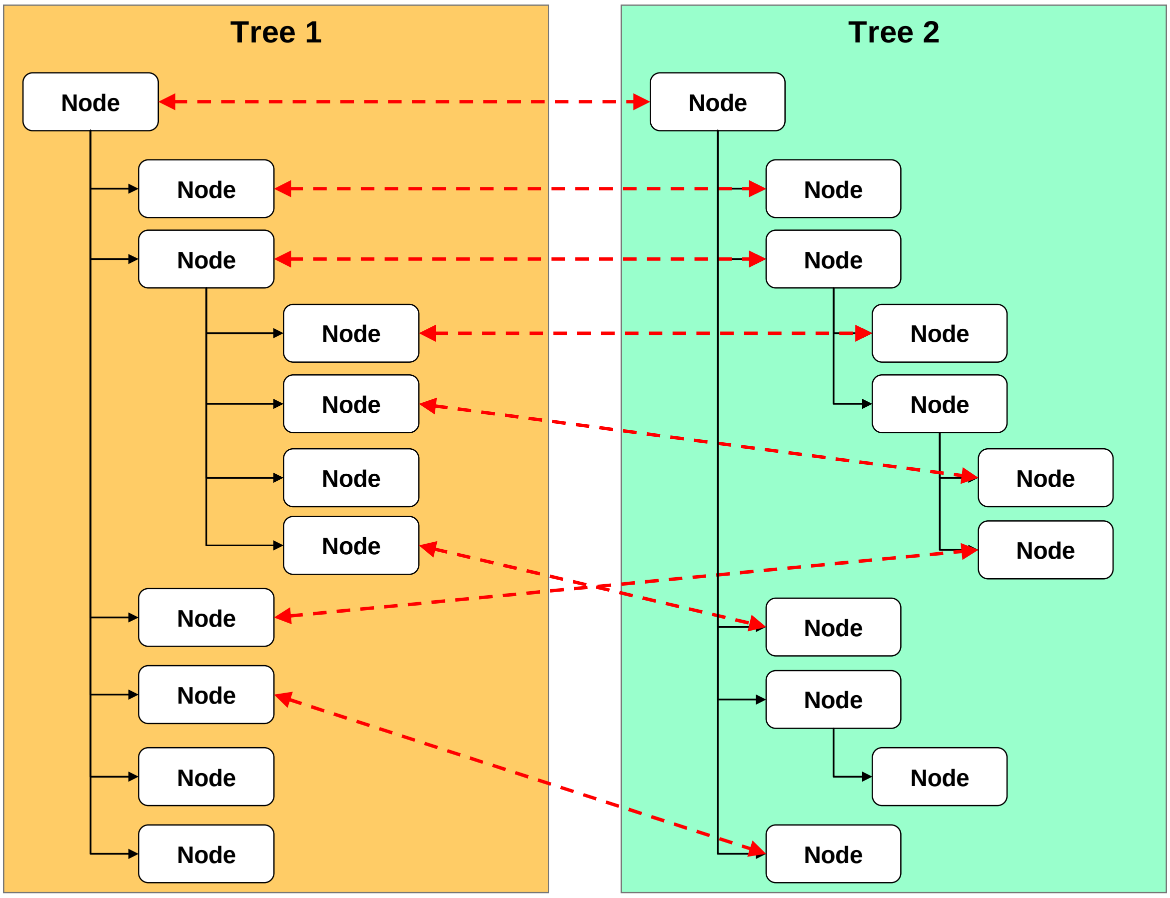

Figure 1: Data structure difference

Because of the similarity of both trees it is possible to define a mapping between both of them. There are several cases how the nodes have to be mapped:

- node x of tree 1 is equivalent to node y of tree 2 and also the other way around
- node x of tree 1 is equivalent to a path of nodes through tree 2
- a path of nodes through tree 1 is equivalent to node y of tree 2

It is not necessary to define mappings for nodes whose semantic meaning or information is not carried by the respective other tree.

CATALOGUE DATABASE

Catalogues need a database for the information provided by them. For each request received by any catalogue interface this database must be queried and the requested information transformed into the data structures of the protocol used by the interface. The data can then be returned to the client. For effectiveness the requests of a client must be transformed as directly as possible into queries on the database. Further the transformation of the result data into protocol specific structures should be as fast and as directly as possible.

In the previous chapter it is shown how different data structures can be mapped to each other. If a catalogues database defined a data structure similar to a tree, it should therefore be possible to define also a direct mapping of this structure to a protocol specific data structure. This chapter will therefore describe how models for metadata stored in catalogues should be defined and how the metadata following those structures could be represented physically for example in a database. Based on this it will also be shown how order options could be stored as well using the same principles.

Metadata Models

As shown in the previous chapter metadata models for earth observation data can be seen as trees. Such trees can be modelled object oriented, where each node is an object which refers to other objects as its children. Therefore object oriented approaches are applicable for defining metadata models. In this case there must be first of all an object oriented definition of the metadata structures. Instances of such definitions, the objects, are then for example a set of metadata for a concrete earth observation product. An object oriented model for the metadata structure of an earth observation product could look like shown in Figure 2. This figure does not make use of the unified modelling language, but shows at least a simple structure for earth observation metadata.

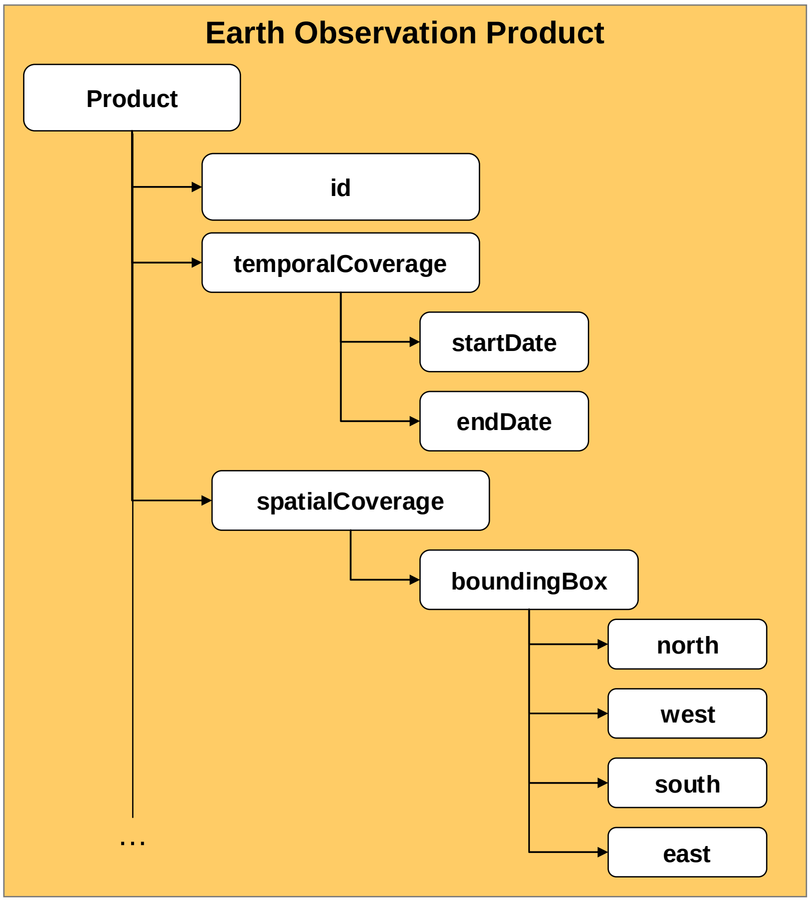

Figure 2: Simple metadata structure for EO products

The next challenge is the representation of such structures in a data base system, leading over to the next chapter.

Metadata Representation

The metadata structures defined so far have an object oriented approach. The instances of such structures, being the metadata of existing EO products, needs to be stored in databases. Object oriented databases could be easily used for storing such structures. There are also databases based on the eXtensible Markup Language (XML) being able to carry object oriented structures as well. Further there is the possibility of using relational databases together with technologies allowing an object-relational mapping based on existing technologies. All those database types have one thing in common: They need an object oriented definition, i.e. a model, of the structures that must be stored. Such a definition could be metadata models as shown in the previous section. Based on these models the different data base forms create internally a physical representation of the metadata. In the case of relational databases with an object-relational mapping technology this representation would be a set of tables with reference columns.

Independent of the used database approach it is therefore possible to store object oriented metadata structures and access them either using methods provided by the data base technology or by directly reading the physical representation, which is easiest with the relational databases.

Order Option and Order Storage

Order options for earth observation products have models as well. Usually those models are similar to the object oriented models for product metadata. Therefore the general approaches for defining such models and storing the instances of those models could also be applied to order options. This leads to the fact that order options could be modelled and accessed as directly as possible with the used database technology as well. This also applies to orders for earth observation products.

ACCESS FUNCTIONALITY GENERATION

If there exist model definitions for the data structures stored in a catalogue's database as well as for the data structures of a specific protocol, it is also possible to define a mapping between those structures. Based on these two models and their mapping it is then possible to generate access methods being able to access the data inside the database and return it in the structures requested by the specific protocol. This generation process is more detailed described in this chapter.

Access Method Requirements

There are two major functionalities required by access methods being able to access metadata and return it in a protocol specific structure:

- searching for data based on protocol specific conditions
- returning matching data in the protocol specific data structures

The first point can be subdivided into the following sub functionalities:

- parse the protocol specific search conditions provided by the client
- map those conditions to equivalent conditions being executable on the database
- execute the query on the database
- map the query result from the database specific structures to protocol specific result structures

The last of these points could be equal to returning matching data to the client in protocol specific data structures. Anyway there are protocols dividing between data searching and retrieving the results. With such protocols the result of a search could be only the number of data sets, matching the query conditions. Later in a second request the clients request a subset of the found data sets to be retrieved from the database and then to be returned in protocol specific structures as the result of this request.

Independent if the protocol returns only the number of records or directly the found data as a reply for a search, the two functionalities condition mapping and result mapping must be supported by the generated access methods.

General Generation Mechanisms

Protocol specific access methods might now be generated based on the model definition of the database, the model definition of the protocol and the mapping definition between both models. A generator for such a generation process must perform the following steps:

- parse the model describing the data structures of the database
- parse the model describing the data structures of the specific protocol

- parse the mapping between both data structures
- generate access methods being able to map query conditions defined in the specific protocol to conditions for the database
- generate access methods being able to retrieve data from the database while directly creating protocol specific data structures

The query conditions mapping methods must implement a parsing algorithm, which is able to extract the several conditions from protocol specific queries defined by clients. For each of these conditions it creates at runtime at least one semantically equivalent condition in a syntax which is processible by the catalogues database. These conditions are then combined to a query being executed on the database. The result of this query is then parsed by the access methods. This process directly creates the protocol specific structures to be returned to the client and fills them with the information resulting from the parsing. (Hint: This result must not directly be the data set found by the executed query. It could also be just the number of records found, as it is for example in CIP. The data sets could be returned as a result of later requests of the client. Anyway the mapping between the found data sets to a protocol specific structure is identical.)

Access Methods for Product Metadata, Order Options and Orders

The generic principles for protocol specific access method generation described in the previous section are applicable if the following three statements are matched:

- the data structures in the catalogue's database have a defined model
- the protocol used for accessing the data has a defined model for its data structures
- a mapping between both models is defined

Therefore these principles can be adapted to product metadata, order options and also to orders in the earth observation domain.

APPROACH OF A CATALOGUE

After having shown how data structures can be modelled, how they can be mapped to each other and how methods can be generated being able to accept requests relying on one model, accessing data being defined in another model and returning results again based on the expected model, it is now possible to implement a catalogue having one database but providing several different interfaces. The basics of such implementations are described in the following.

Protocol Specifics

Interface protocols provided by earth observation data catalogues do not only define the searching for and the retrieval of product metadata. Beneath this they do also define further general protocol behaviour as listed in the introduction of the section "Catalogue Interfaces". Therefore the provision of a specific protocol by a catalogue is not only implemented by generating protocol specific access methods. There need to be also a protocol specific implementation handling for example authentication and authorisation as well as order submission. This implementation therefore encapsulates the access methods, to assure that the catalogue fully implements the functionality of a protocol. The combination of these implementations, the access method generators and the created access methods results in the generic structure for earth observation data catalogues shown in Figure 3.

In this figure first of all the common database of the catalogue ("Catalogue's Database") and the model for the data structures in this database ("Catalogue Data Structure Model") is shown. Further there is an "Access Method Generator" taking the "Catalogue Data Structure Model", the "Protocol Data Structure Model" as well as the mapping between those two models ("Model Mapping Definition") and generates based on this input several protocol specific "Access Methods". Those "Access Methods" are then utilized by the protocol specific implementation ("Protocol Interface") to provide a specific protocol interface for the catalogue. The figure shows only one generator but three different interfaces provided by the catalogue. It could also be the case, that there are three different generators, one for each interface. But this is a design decision of the catalogue implementer.

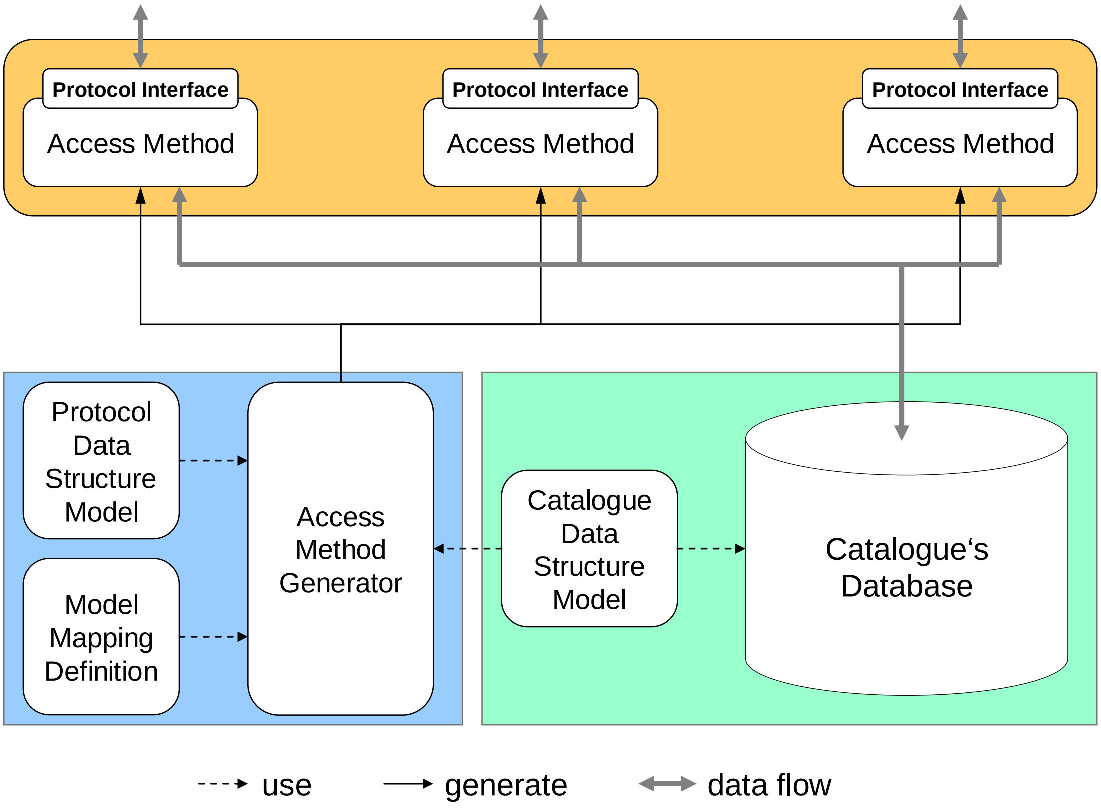

Figure 3: Generic structure for earth observation data catalogues

Metadata, Order Option and Order Storage

The generic approach introduced in the previous section is applicable for accessing product metadata, order options as well as orders. The reason for this is that all these types of data can be modelled in the way shown in section "Catalogue Database". For orders some extensions are needed, as they require a processing after having been submitted to the catalogue. But this only means that the protocol specific implementations combining the access methods to complete interfaces must be more complex. They need for example to be able to validate orders, forward them to the appropriate processing systems, handle order states, etc.

EOWEB® AS A REFERENCE IMPLEMENTATION

The Earth Observation on the WEB (EOWEB®) portal of the German Remote Sensing Data Centre (DFD) of the German Aerospace Centre (DLR) is a proven implementation of the concepts shown in this paper. EOWEB® is a catalogue for earth observation data products based on the Catalogue Interoperability Protocol (CIP). It has the following layers being mentioned from top (client side) to bottom (server side):

- EOWEB® Gateway – providing web interfaces for users accessing the catalogue
- EOWEB® Retrieval Manager – the protocol specific implementation of the catalogue
- EOWEB® Dataserver – protocol specific access method generator together with the generated access methods

- EOWEB® Product Library – database system of EOWEB® based on the Product Library Software of the DFD

Although EOWEB® only supports CIP so far it is designed based on the generic structure for earth observation data catalogues as shown in Figure 3 with the difference that so far only one protocol specific generator has been implemented which creates only CIP specific access methods. Anyway this could be extended to support also further protocols without the need of changing the database for the catalogue. The diverse layers of EOWEB® refer to Figure 3 as follows:

- EOWEB® Gateway is not shown in Figure 3 as it is a protocol specific client of the catalogue
- EOWEB® Retrieval Manager is equivalent to the box named “Protocol Interface”
- EOWEB® Dataserver is the combination of the box named “Access method” and the box “Access Method Generator”
- EOWEB® Product Library is equivalent to the combination of the boxes “Catalogue Data Structure Model” and “Catalogue’s Database”

From the different forms of databases applicable for such catalogues (as listed in “Metadata Representation”) EOWEB® utilizes the one with a relational database in combination with an object-relational mapping which is in this case provided by the Product Library software. At runtime EOWEB® parses CIP requests and maps them directly to SQL statements being executed on the relational database of the Product Library. The results of these statements are then put into CIP messages which are sent back to the client.

EOWEB® also proves the concept of having several separated interfaces on one database. In the EOWEB® test environment for example exist at least two access method generators, therefore two parallel generated access methods as well as two parallel protocol interfaces. Both refer to the same EOWEB® Product Library as their database. If a new product metadata set is inserted into the EOWEB® Product Library, the product can be found via both EOWEB® interfaces. If an order is submitted via one of the interfaces, it can directly be monitored using the other interface. The same order options are referred by the two instances as well.

TECHNOLOGY DISCUSSION

As already shown the concept is able to support a broad variety of protocols and can be based on different database technologies. Considering performance and robustness the approach with a relational database should be preferred as these databases have a longer history than the object oriented ones. The relational database of EOWEB® is the commercial Informix Dynamic Server with the Geodetic Data Blade as plug-in to allow storing spatial coverages and performing spatial searches. This could be exchanged by the PostgreSQL database management system with a plug-in for spatial data. This plug-in could be PostGIS or the PostgreSQL extension implemented by Werum Software & Systems called GeoPG (Geographical extensions for PostGreSQL).

EOWEB® itself is a J2EE (Java Enterprise Edition) application running in a JBoss application server. This is not a mandatory approach. Catalogue implementations could also be implemented J2EE and JBoss independent or utilizes other technologies like other application servers or frameworks.

The complete concept is multi mission capable. The reason for this is that the data models for the database, as well as most data models of provided protocols are extensible to allow the configuration of new product types. When using the object oriented derivation of models it is also possible to define base models for earth observation data products, which can be reused as base for any newly defined product type.

CONCLUSION

The described concept allows for setting up catalogues

- having one data base
- allowing data provision via many different protocols based on the same data base
- being multi mission capable
- being extensible to support further interfaces
- being technology independent

The concept is proven by the existing implementation of EOWEB®. It could therefore be a basis for the development of future catalogues for earth observation data products.

REFERENCES

[1] – P. Harms: Optimierung von Zugriffsmethoden auf objektorientierte Metadatenmodelle von Fernerkundungsdaten für den Zugriff über CIP (Catalogue Interoperability Protocol) (eng: Optimisation of access methods on object-oriented meta data models of earth observation data for the access via CIP).Diploma Thesis, Werum Software & Systems AG (2004)

[2] – Earth Observation on the WEB (EOWEB®), earth observation data catalogue of the German Remote Sensing Data Centre (DFD), www.eoweb.de

[3] – CEOS - Working Group on Information Systems and Services - Protocol Task Team: Catalogue Interoperability Protocol (CIP) Specification – Release B. (2004)

[4] – S. Kiemle. From Digital Archive to Digital Library – A Middleware for Earth-Observation Data Management. In 6th European Conference on Digital Libraries (ECDL), Rome, Italy, pages 230-237, Sept. 2002.

I want to thank my collegues at Werum Software & Systems AG for supporting me during my diploma thesis on this topic as well as during the implementation of EOWEB®. Many thanks also to the responsibilities of DLR for contracting Werum to implement the EOWEB® backend. Special thanks go to Daniele Dietrich, Stephan Kiemle and Anne-Katrin Schröder-Lanz of the DLR for supporting me during the development and implementation and for answering all questions I had immediately and with competence, no matter how weird they might have been especially at the beginning.