From Digital Archive to Digital Library – A Middleware for Earth-Observation Data Management

Stephan Kiemle

German Aerospace Center (DLR)
German Remote Sensing Data Center (DFD)
Oberpfaffenhofen, D-82234 Weßling, Germany
Stephan.Kiemle@dlr.de
http://www.dfd.dlr.de

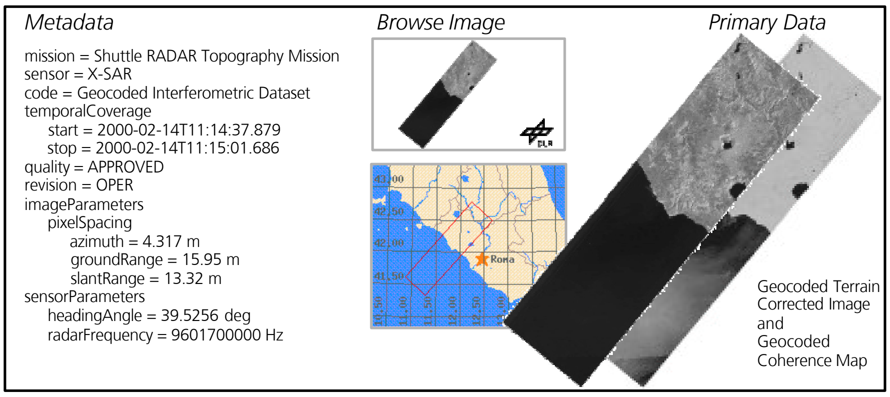
Abstract. The German Remote Sensing Data Center (DFD) has developed a digital library for the long-term management of earth observation data products. This Product Library is a central part of DFD's multi-mission ground segment Data Information and Management System (DIMS) and is successfully in operation since 2000. Its data model is regularly extended to support products of upcoming earth observation missions. The Product Library implements a middleware filling the gap between application-level object data models and physical storage structures such as a digital robot archive with hierarchical storage management. This paper presents the principles of the Product Library middleware and its application in the specific earth observation context.

1 Introduction

Digital libraries are generally based on powerful catalogue and archive systems for the management of metadata and primary data. The semantic gap between high-level information modeling, search and retrieval requirements on one side and existing data storage systems on the other side requires keen solutions to overcome nonsatisfying compromises with respect to the sustainability of the library system.

In the context of earth observation where hundreds of terabytes have to be kept accessible for long term, the sustainability of a digital library is a key issue. The library has not only to cope with a permanently growing amount of data, diversity of data and evolving physical storage technology, it has also to integrate existing digital archives and provide application-level interfaces for other services in order to jointly cover all earth observation ground system tasks of product processing, monitoring, storage, ordering and delivery [1].

The basic requirement of sustainability led to two main concepts followed by the Product Library: first the separation of metadata characterizing a product and the original primary data of the product, and second a middleware decoupling the application interface from the internal storage solution. Both ideas are principally applicable to other digital libraries handling large data objects such as image, audio and video data.



```
mission = Shuttle RADAR Topography Mission
sensor = X-SAR
code = Geocoded Interferometric Dataset
temporalCoverage
    start = 2000-02-14T11:14:37.879
    stop = 2000-02-14T11:15:01.686
quality = APPROVED
revision = OPER
imageParameters
    pixelSpacing
        azimuth = 4.317 m
        groundRange = 15.95 m
        slantRange = 13.32 m
sensorParameters
    headingAngle = 39.5256 deg
    radarFrequency = 9601700000 Hz
```

Fig. 1. Earth observation product example with metadata, browse and primary components. The scene shows a geocoded RADAR amplitude image of the tyrrhenian coast north of Roma

A typical earth observation data product consists of different components like the original primary data (e.g. hierarchical data format files), browse images reduced in resolution and auxiliary data useful e.g. for data interpretation (Fig. 1). In addition, each product is described by metadata, parameters describing the product like geo-temporal coverage information, sensor adjustments and feature vectors. Metadata is typically used to search, identify, retrieve and analyze products. The separation of metadata and primary data is motivated by the huge difference in size. Whereas metadata usually takes several Kbytes, the primary data can have up to a GByte of size, depending on the geographic extension, resolution and dimension (e.g. number of channels) of the data.

The Product Library manages products as a whole but internally separates metadata and primary data to store it in an inventory and an archive service [2] (Fig. 2). The inventory service is optimized for search and retrieval of geo-referenced metadata and based on an object-relational database management system. The archive service provides management of primary product files based on a hierarchical storage management system with extensible capacity[1].

The consistency between archive and inventory part is guaranteed by a transaction mechanism spanning both parts controlled by the Product Library itself. The internal Product Library inventory and archive services are implemented in Java and interact through CORBA [3], like all other services of the Data Information and Management System (DIMS).

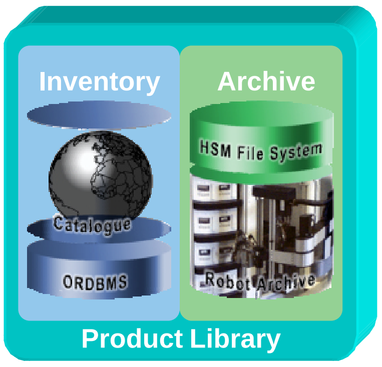


Fig. 2. Product Library Structure

[1] Currently the DFD Product Library instance manages more than half a million product items and the archive hosts 25 TByte, numbers which are expected to multiply by ten until 2005.

2 Metadata Modeling

The Product Library inventory service is responsible for the management of product metadata and provides a catalogue with the capability of insertion, update, versioning, retrieval and deletion of product metadata. The catalogue is configured with a UML object data model defining the structure and relation of metadata items representing the earth observation products. The central part of this model is the basic product model defining the main generic earth observation product classes and their relationships (Fig. 3). A product, which can be bundled in product groups, aggregates primary data components. Products, groups and components can be generalized as product items with common features like creation time, spatio-temporal coverage and aggregated browse data. Each product item belongs to a collection describing the type of the earth observation item.

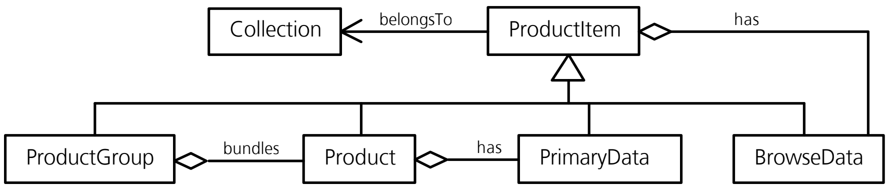


Figure 3. Basic product model (extract)

Each time new products have to be supported, the product model configuration is extended by new specific definitions like product types, product component types and product parameters. Therefore the generic classes *ProductGroup*, *Product*, *PrimaryData* and *BrowseData*, or other previously defined classes are extended by new specific classes. Specific parameters can be added by defining new attributes. Attributes that have already been defined for other types can be reused in the definition of new types. This is possible because the inventory service stores all modeling elements (classes, types, attributes, methods, references) ever defined in a meta object repository[2].

The Product Library provides a configuration tool for the definition of metadata model extensions. This tool has direct access to the meta object repository so that the administrator is able to browse the modeling elements that have already been defined. The administrator is encouraged to reuse definitions as much as possible to avoid an uncontrolled growth of modeling elements with similar semantics. After configuration of a UML metadata model extension, the inventory service automatically deduces mapping rules that map the different modeling elements to relational structures. The metadata model extension requires no software updates and can be performed during Product Library operation.

By applying the generated mapping rules, the inventory middleware decouples the application metadata view from the relational storage organization. Tables can be

[2] The modeling elements and their relationships are defined in the so-called MetaItemModel, a UML data model describing the modeling capabilities which is itself defined in the meta object repository. The meta object repository is therefore self-contained.

renamed, fragmented, split, indexed and merged for performance or other internal reasons without effect on the metadata model presented at the Product Library interface. In addition new metadata views can be defined in parallel upon the same underlying relational structures e.g. to respond to specific catalogue protocol requirements of a certain application client. This feature becomes more and more important in the context of interoperable library systems and integration with geographical information systems.

At the external Product Library interface, the product structure and metadata is represented in an XML document, the so-called item information file. This document has to be provided if a product shall be inserted and it is newly generated by the Product Library each time a product is extracted. For each product item, i.e. the product itself and each component, the item information file specifies

- the type of the item,
- its logical identifier, a list of specific attributes uniquely identifying the item,
- the aggregated components,
- the location (name, path, host) of the data files,
- the basic product item metadata (applicable to all product items) and
- the specific product item metadata.

The following paragraph shows an extract of the XML document type definition (DTD) of the item information file. Aggregated components and other associated items can either be referenced locally within the item information file or reference existing items in the Product Library by indicating their identifier. All structures within the item information file are well defined except the specific product item metadata. Here, the DTD allows any structured, list and set parameters because the specific metadata structure changes from one product type to another. (Element nodes not further detailed are text nodes mapping to #PCDATA.)

```
<!ELEMENT IIF (item*)>
<!ELEMENT item (admin,components,fileInformation,
                parameters,specificParameters)>
  <!ELEMENT admin (type,keys)>
    <!ELEMENT keys (basicFeature*)>
      <!ATTLIST basicFeature name CDATA #REQUIRED>
  <!ELEMENT components (component*)>
    <!ELEMENT component (role,ref)>
      <!ELEMENT ref (localId|remoteId)>
  <!ELEMENT fileInformation (file*)>
    <!ELEMENT file (host,path,name)>
  <!ELEMENT parameters (availability,quality,creation,
            spatialCoverage,tempCoverage,predecessor)
    <!ELEMENT creation (time,creator)>
      <!ELEMENT creator (remoteId)>
    <!ELEMENT spatialCoverage (point|box|circle|
                               string|polygon)>
      <!ELEMENT point (latitude,longitude)>
      <!ELEMENT box (east,north,west,south)>
      <!ELEMENT circle (point,radius)>
      <!ELEMENT string (point*)>
      <!ELEMENT polygon (point*)>
    <!ELEMENT tempCoverage (start,stop)>
    <!ELEMENT predecessor (remoteId)>
  <!ELEMENT specificParameters (feature*)>
    <!ELEMENT feature (#PCDATA|feature*)>
    <!ATTLIST feature name CDATA #REQUIRED>
```

3 Archive Management

The primary data of earth observation products usually consists of a couple of files per product component, the smallest logical unit managed by the Product Library. In order to decouple and hide internal storage issues, all product files have to be managed by the archive middleware such that

- the internal archive location is hidden to the external user
- the internal archive location is comprehensive to the internal archive operator (concise root points, balanced directory trees etc.)
- the internal archive location fulfils the requirements of the underlying archive system for storage/access optimization (e.g. files spread over different media types)
- the files can be extracted again given the identification of the product component previously retrieved from the inventory service.

The archive service of the Product Library is structured in archive area sub-services, one area per file system. The archive area service is responsible for the file I/O (via FTP) to and from its archive file system which can be a standard UNIX file system or a hierarchical storage file system [6] where certain file I/O activities are optimized to avoid unnecessary media reload and positioning procedures. The underlying robot library is able to handle different types of media, for example magneto-optical discs and tapes from different vendors. Several media drives fed by the robot operate in parallel. When new data is copied on the cache, the robot library automatically generates a first and a second copy on media. For safety reasons the second copy is not stored in one of the library towers but off-line in a separate location and is used only for disaster recovery.

The middleware implemented by the archive service maps primary product file requests like insertion, retrieval and deletion to the corresponding archive area and directory path. Therefore, the archive service is configured with the archive mapping rules consisting of

- the product component type for which this mapping is applicable
- the archive area responsible to store data of this type
- the path rule defining how the directory path within the archive area’s file system has to be built up.

The path rule takes metadata of the product component to generate an individual directory path for the component files. However, the path needs not to be unique because the internal file names contain the unique identifier of the product component to avoid collisions and enable version management. For means of directory path balance, path rules are configured individually using comprehensive metadata parameters of the specific product component.

Table 1. The path rule configured at DFD's operational Product Library for the product component type "MODIS.L0-RAW", the primary data of the level 0 product acquired from the optical sensor Moderate-Resolution Imaging Spectroradiometer

Parameter	Type	Append	*Example Value*
mission	String	_	`TERRA`
sensor	String	_	`MODIS`
code	String	/	`L0`
startTime	date, format: y_m/d		`2002-05-13T16:09:12.281`

Assuming a parameter value assignment with the example values in the table, the resulting directory path will be `TERRA_MODIS_L0/y2002_m05/d13` and the component files will be transferred into this directory relative to the archive area root location, usually the file system mount point. As shown with the mapping of the *startTime* parameter, the archive service uses the configured format definition and performs certain modifications to guarantee results usable for directory names (e.g. elimination of special characters).

For all earth observation product components inserted in the Product Library, the internal archive location is determined in this way. For consistency purpose this internal location is additionally stored in a special space within the inventory service, the file information catalogue. This catalogue is exclusively accessible for internal use and not part of the Product Library application interface. The archive operator can browse the file information catalogue e.g. to get an overview about the internal archive structure or to search for specific files.

Products that are already stored in an existing archive file system can be registered whereby the library takes over their management, adding their file locations to the file information catalogue.

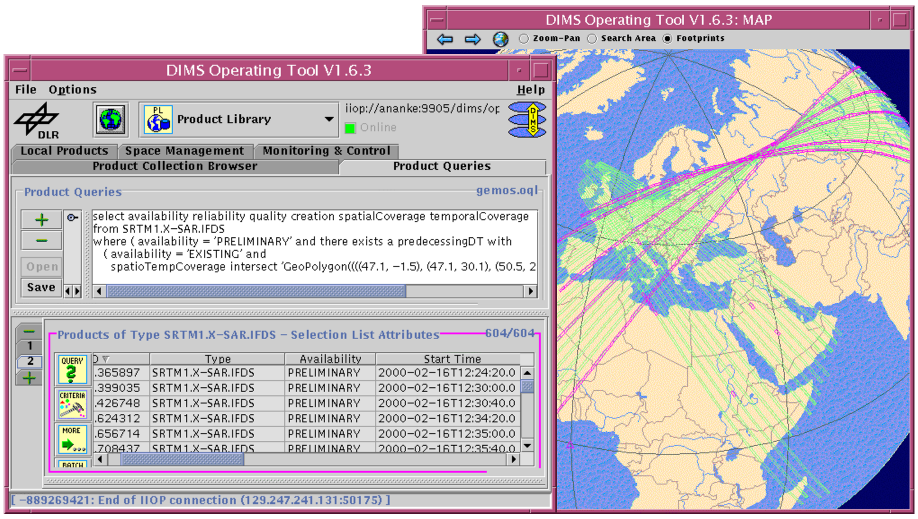
The Product Library uses itself the file information catalogue when products have to be delivered. The external user or system specifies a product identifier and the archive service middleware takes this identifier to look up the internal file location in the file information catalogue. With this information it can request the file delivery from the responsible archive area.

By applying archiving rules, the archive middleware of the Product Library hides internal storage structures. Advantages are e.g. internal migration of data to future media generations, reorganization of archive hosts, file systems and directory paths without effect on external Product Library client systems.

4 Query Interface

In general the query requirements of operators and client services like processing systems are not predictable. Therefore, the Product Library provides an ad-hoc query interface and an object query language which is a subset of the Object Data Management Group OQL standard [5] extended by special features for the earth observation context such as search methods like intersect, inside, outside, within and beyond for spatio-temporal specifications.

In the Operating Tool, the graphical operator client of DIMS, the operator can textually enter any OQL query and submit the query to the Product Library. The inventory service maps the query to SQL queries processed by the underlying database system. The Operating Tool visualizes query results in a result table, a detailed metadata view (on demand) and on the map.



```
select availability reliability quality creation spatialCoverage temporalCoverage
from SRTM1.X-SAR.IFDS
where ( availability = 'PRELIMINARY' and there exists a predecessingDT with
   ( availability = 'EXISTING' and
     spatioTempCoverage intersect 'GeoPolygon((((47.1, -1.5), (47.1, 30.1), (50.5, 2
```

D	Type	Availability	Start Time
.365897	SRTM1.X-SAR.IFDS	PRELIMINARY	2000-02-16T12:24:20.0
.399035	SRTM1.X-SAR.IFDS	PRELIMINARY	2000-02-16T12:30:00.0
.426748	SRTM1.X-SAR.IFDS	PRELIMINARY	2000-02-16T12:30:40.0
.624312	SRTM1.X-SAR.IFDS	PRELIMINARY	2000-02-16T12:34:20.0
.656714	SRTM1.X-SAR.IFDS	PRELIMINARY	2000-02-16T12:35:00.0
.708437	SRTM1.X-SAR.IFDS	PRELIMINARY	2000-02-16T12:35:40.0

[-889269421: End of IIOP connection (129.247.241.131:50175)]

Fig. 4. Ad-hoc product query interface in the DIMS Operating Tool

The example in Fig. 4 shows an object query that searches all preliminary interferometric data sets whose associated predecessing data take is existing and intersects a certain region over central Europe. The footprints of the resulting products are shown on the map (in pink). For additional information all data takes intersecting the same region have been queried before (displayed in green).

The Product Library query interface is not only used by operators but also by processing systems e.g. to find the correct input products required for the processing of certain output products. Processing systems can also subscribe for specific product events, specifying the action (e.g. product insertion) and a query condition that must be fulfilled by the corresponding product. This triggering mechanism is used to build up self-controlled processing chains where value-added products are automatically generated when the required low-level input products become available.

5 Conclusion

The Product Library is a live example that a transition from existing digital archives to digital library systems is feasible and brings comprehensive data management functionality and finally increases the value of the archived data e.g. by adding sophisticated search and retrieval capabilities. The library middleware decouples the application oriented external interface from internal storage concerns and therefore guarantees stable interfaces and reduces administration efforts – precondition for sustainable long-term management of earth observation data products.

The archive service middleware determines where and how primary data files have to be inserted in/extracted from the archive system. Therefore, an external application system does not need any knowledge about the physical archive structure like file systems and directory paths. Existing archive file systems can be integrated and reorganized e.g. for migration purpose without impact on the application interface.

The inventory service middleware allows easy and flexible application object data modeling and maps object models to relational structures. Based on this mapping, the inventory automatically translates catalogue search queries specified in the expressive application-level object query language. Again the physical storage can internally be optimized without impact on the library interface, easily overcoming the minimal performance impact of the mapping process.

The Product Library middleware is implemented in Java and uses state-of-the-art technologies and common standards like CORBA, XML and JDBC [4]. It is one of the services of the Data Information and Management System (DIMS) and operational since 2000, hosting more than half a million product items (25 TByte data) of various airborne, shuttle and satellite missions.

References

1. Mikusch, E., Diedrich, E., Göhmann, M., Kiemle, S., Reck, C., Reißig, R., Schmidt, K., Wildegger, W., Wolfmüller, M.: Data Information and Management System for the Production, Archiving and Distribution of Earth Observation Products. Data Systems in Aerospace 2000, EUROSPACE. ESA Publications Division, SP-457, Noordwijk (2000)
2. Kiemle, S., Mikusch, E., Göhmann, M.: The Product Library – A Scalable Long-Term Storage Repository for Earth Observation Products. Data Systems in Aerospace 2001, EUROSPACE. ESA Publications Division, SP-483, Noordwijk (2001)
3. Object Management Group and X/Open: The Common Object Request Broker 2.3: Architecture and Specification. John Wiley & Sons Inc., New York (1999)
4. Kiemle, S., Mikusch, E., Reck, C., Reißig, R., Wildegger, W., Wolfmüller, M.: Data Information and Management System for Earth Observation Products based on CORBA and Java. EOGEO 2000, http://webtech.jrc.it/eogeo2000. JRC, Ispra (2000)
5. Object Data Management Group ODMG: The Object Data Standard: ODMG 3.0. Morgan Kaufmann (2000)
6. Kampa, R.A., Bell, L.V.: UNIX Storage Management. Springer-Verlag, Berlin Heidelberg New York (2002)