# PROCESSING MANAGEMENT TOOLS FOR EARTH OBSERVATION PRODUCTS AT DLR-DFD

**M. Böttcher[(1)], R. Reißig[(2)], E. Mikusch[(2)] , C. Reck[(2)]**

*[(1)] Werum Software & Systems, D-21337 Lüneburg, Germany, phone: +49 4131 8900-84*
*e-mail: boettcher@werum.de, www: http://www.werum.de/*
*[(2)] German Aerospace Center (DLR), German Remote Sensing Data Center (DFD)*
*Oberpfaffenhofen, D-82234 Weßling, Germany, phone: +49 8153 28 2721*
*e-mail: eberhard.mikusch@dlr.de , www: http://www.dfd.dlr.de/*

## ABSTRACT

Processing earth observation data is not only an algorithmic issue. It also raises questions of integration and long term operation in a processing, archiving and distribution facility. This paper presents a kind of middleware to support integration of existing and newly developed data processing software into the DIMS operational environment. DIMS is the multi-mission Data and Information Management System, a processing and archiving and user service facility developed at the German Remote Sensing Data Centre DLR-DFD.
During the next years more than 50 processing systems for earth observation data are to be integrated and set up for operational service in DIMS. In this paper we argue that integration of data processing software shall not only map interfaces. Instead, common functions used in all or at least several data processing systems shall be provided as tools, utilities and framework. This reduces complexity for processor developers, it increases uniformity for the operational phase, and it reduces costs by reuse of software components for current and future missions.

## 1. THE CHALLENGE

During the next years more than 50 processing systems for earth observation data are to be integrated into the Data Information and Management System (DIMS) at DLR-DFD. DIMS is a multi-mission processing and archiving and distribution facility for earth observation products [1] (Fig. 1).

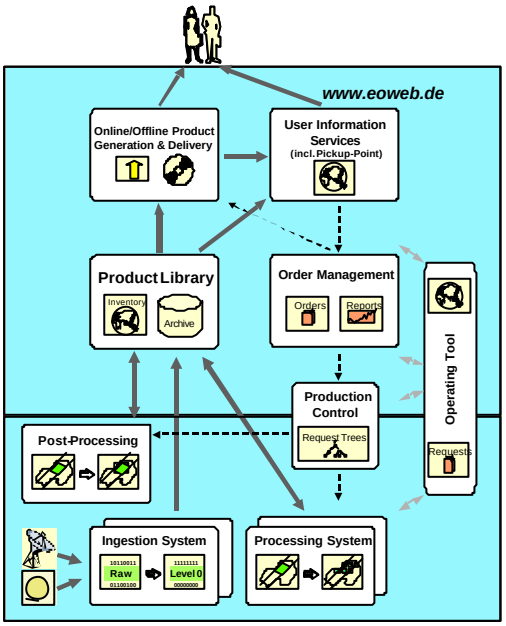


Fig. 1: Processing Systems in DIMS (functional view)

DIMS is a development of DFD. Processing systems developed by various research groups are to be integrated successively. They will be operated for years. Thus, it seems to us to be crucial that operating is efficient and

homogeneous. And it can be expected that the processing systems will themselves go through the usual software evolution processes.
Experiences from earlier archiving system developments let us decide to provide tools and frameworks for processing systems right from the beginning of DIMS development. These tools are the DIMS Processing System Management (PSM) and the DIMS Operating Tool (OT). Technically speaking, the PSM provides an interface between the implemented processing algorithms on one side and the processing and archiving facility of DIMS on the other. At the DIMS side, two components are involved: DIMS Production Control that controls what to process on which processing system, and DIMS Product Library that serves input products and receives processing results for cataloguing, long term archiving or short term storage.
Processing systems are heterogeneous. They vary in programming languages and platforms used, and they range from simple one step input-output systems to complex distributed multi-step systems with user interaction and elaborated workflow control. Examples of complex cases are among those for the Shuttle Radar Topography Mission (SRTM) [2] or for the Global Ozone Monitoring Experiment (GOME). Simpler cases are the raw data processing for the Challenging Minisatellite Payload mission for gravity and magnetic field measurements (CHAMP). But there is also a lot in common among these systems, and that is what we address with the tools.
For a family of products ranging from raw data received from a downlink up to high level or value added products there usually is a corresponding family of processing systems that process them. What is common to almost any family of products is that the scenarios change over time: first product "ingestion" of low level products where the inputs come from sources outside of the DIMS Product Library, then systematic processing of some levels, finally production on (user order) request only, and post-processing. All those scenarios are supported by DIMS and the PSM.
Not only processing systems but also their developers are heterogeneous. It is a task to introduce certain integration procedures or guidelines. They shall help to mediate between processing algorithm know-how and the requirements of integration and operation. We do not have stable experiences with the establishment of procedures and guidelines up to now. But we know that they are necessary. This "standardisation" issue is also supported by the fact that sometimes systems tend to reside in an institution longer than their developers. So, it is a challenge to support a modular processing system design and a homogeneous user interface, by tools and by procedures. We concentrate on the tools in this paper.

## 2. PROCESSING SYSTEM MANAGEMENT – MORE THAN AN INTERFACE

A processing system that shall be integrated with the DIMS processing and archiving facility has to use and implement certain interfaces and protocols. The processing algorithms of a processing system are implemented in one or more processors. With the DIMS Processing System Management (PSM) to be used in any processing system, we introduce another layer between the processing algorithms and DIMS. It might be argued that this only moves the interfaces for processor developers a layer downwards. The DIMS Product Library [3] has an interface for products as well, and DIMS Production Control has one for production requests. Does the interface get simpler this way? This is the case, indeed! It is more local and restricted. It handles products as files and directories, not as remote "items". And it only deals with a single request at a time, since the multi-request capability is provided by PSM without mapping it to the interface for the processor developer.
But the simpler interface was not the main intention for the introduction of another layer. Instead, the intention was to implement and provide once and in a uniform way, what else would have been implemented by many processing system developers individually. Fig. 2 lists those common functions that we have identified.

- ***Multi-request capability with production request queuing***
- ***Input product retrieval from and output product transfer to the DIMS Product Library***
- ***Visualization and control of a processing system with graphic user interfaces***
- *Autonomous operation capability with product cache management and pre-caching of input products*
- *Rule based workflow control for production request processing*
- *Processor charging, scheduling, load balancing for multiple processors*
- *Persistence and reliability*
- *Support for raw product ingestion from external sources*
- *Support for systematic processing with timers and automatic triggers*
- *Utilities for logging, request archiving, XML*

Fig. 2: How PSM and OT provide common functions for processing systems

The common functions are provided by the PSM and the DIMS Operating Tool (OT). These components are generic, they form a framework that is just configured for individual processing systems. The processing systems get the functions provided without extra efforts.

The first three functions in Fig. 2 are request handling, product handling and operating. They are considered as the most basic ones, required by any processing system in the DIMS environment. The third function is further illustrated in Fig. 3. The DIMS Operating Tool with its views of queues and entries visualises and controls scheduling and processing.

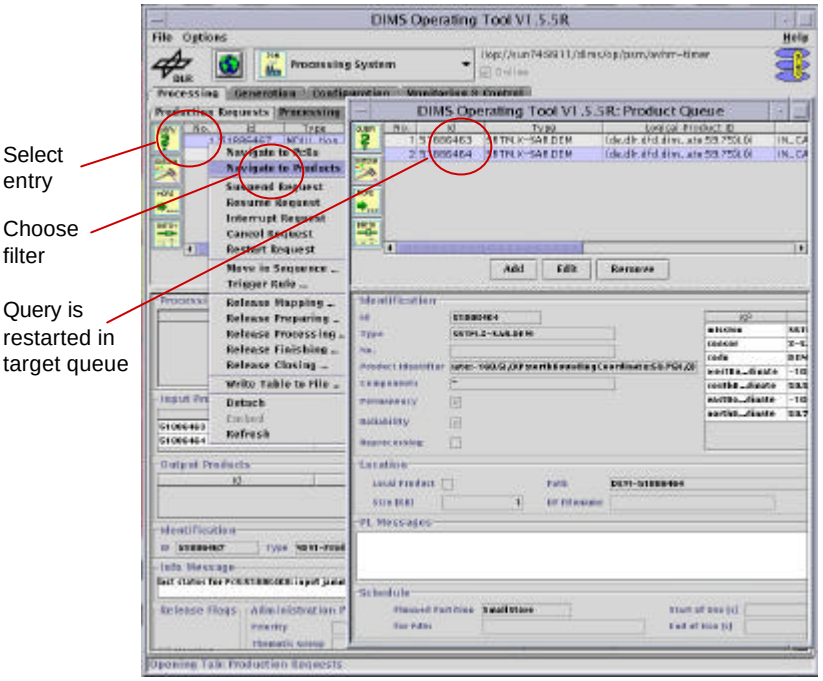


Fig. 3: Queue and entry browsing and control with the DIMS Operating Tool

The seven remaining functions in Fig. 2 are also provided by the PSM. Cache management, scheduling and a number of utilities simplifies the implementation of any processing system. Rule based workflow control with sequences for processing chains, concurrency on multiple data processors and dynamic control for conditions and loops allows for flexibility. With processors as sub-units of a processing system, the PSM supports modularity. Simple processors are often introduced for auxiliary steps, and workflow rules organize their execution. This makes each single processor simpler. With its planning component the PSM performs optimizations. It considers processors as well as cache space as resources. It is not necessary to implement scheduling algorithms in each processing system to achieve load balancing and parallel processing. In many cases processing systems can be scaled by simply adding more processor instances. Persistence of request queues and transaction support guarantees a certain level of fault tolerance.

With the PSM, ingestion processors as well as other processing systems for systematic processing are specially addressed. Fig. 4 displays the variety of scenarios supported.

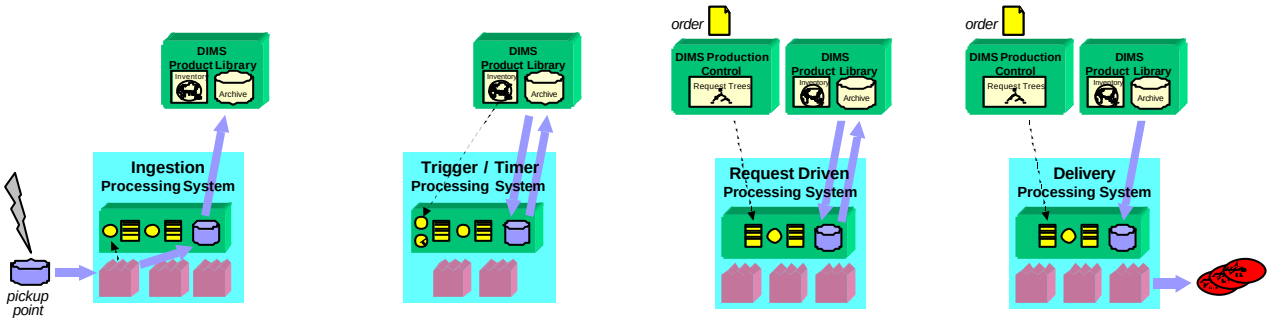


Fig. 4: Systematic and on-demand processing with PSM support

A processing system is configured out of building blocks like timers, automatic triggers, processors, and workflow rules (Fig. 5).

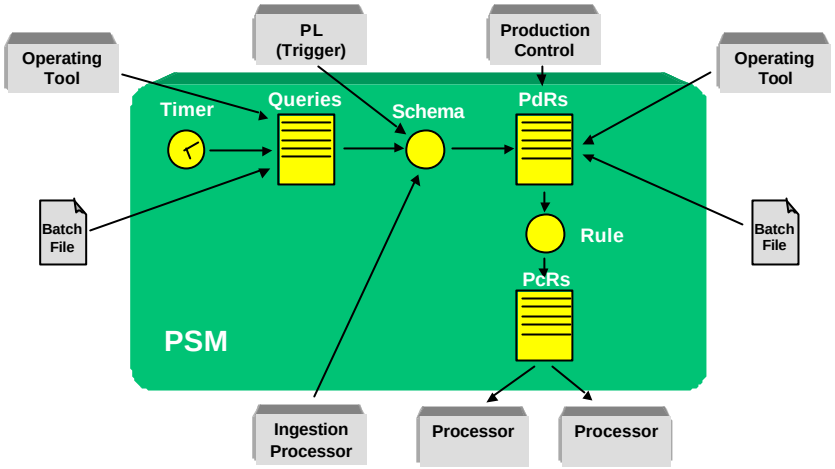


Fig. 5: Production request (PdR) generation in the PSM by timers, automatic triggers and ingestion

A timer can be used to generate a "one-day product" each day. The inputs are determined by a query to the Product Library in this case. An automatic trigger can be used to register for any new product of a certain type. Whenever such a product is delivered to the Product Library by some other processing system, the PSM is triggered and will retrieve the product as input. The cases of timer driven and trigger driven processing as well as raw product ingestion from external sources are mapped to the request driven case for simplicity. Timers, triggers and ingestion processors generate production (or archiving) requests that run through the rule based workflows as usual. The derived processing requests (PcRs) are submitted to processors for processing. By introducing these structures like timers, requests, workflow rules and different kinds of processors it becomes easier for operators and developers to get an overview of a processing system.

## 3. LESSONS LEARNED – EXPERIENCES WITH TECHNOLOGIES

The PSM is written in Java. Java is the core programming language for DIMS. It uses CORBA to communicate with DIMS components. XML is used for configuration, production request description and as product meta data exchange format. It further uses a personalised (single process) edition of an OO-Database for persistence.

Our experiences with Java are that it is sufficiently fast for almost all purposes we use it for. What we really gain of is Java´s robustness against programming errors, its well done integration with GUI programming, and the many software packages e.g. for XML support and other purposes. We have defined a certain framework for DIMS services in Java, and some more generic packages. They complement the many generic packages already found in Java. What we think could be better is its integration with other programming languages.

CORBA is doing its communication task well and reliable. What we suffer a bit from is its lack of standard mechanisms to locate services reliably. The standard name service is not sufficient in a multi server-multi client environment since it requires a certain sequence of registration and lookup (server first). This is not acceptable in case of temporal failure of some service. Thus, we had to implement our extended name service with publisher-subscriber features ourselves.

XML became more and more established as a data description language after DIMS development was started. DIMS uses CORBA data structures in some cases where we nowadays would think about taking XML instead, especially for the exchange of requests. All services do use XML for batch requests already, as well as for the description of configurations. So, the step is not a large one. The advantage is the ability to extend the description without changing old software. The old software will simply ignore the new tags. When sometime this becomes a mayor issue, DIMS may change it.

## 4. CONCLUSION

As usual with generic components and frameworks there is the trade-off between uniformity and flexibility. We have argued for a certain degree of uniformity and modularity. What we provide in return to processing system developers is a set of basic and advanced functions that are required for integration and long term operation of a processing system. What we hope to have preserved is the required amount of flexibility.
The generic components presented in this paper are the DIMS Processing System Management and the DIMS Operating Tool. With them, DIMS implements once and in a uniform way what else would have to be implemented individually by each processing system developer. This saves development efforts, eases maintenance and simplifies operation.

## 5. REFERENCES

1. Mikusch, E., Diedrich, E., Göhmann, M., Kiemle, S., Reck, C., Reißig, R., Schmidt, K., Wildegger, W., Worfmüller, M., *Data Information and Management System for the Production, Archival and Distribution of Earth Observation Products*, DASIA 2000, Montreal, 2000.

2. Roth, A., Eineder, M, Rabus, B., Mikusch, E., Schättler, B., *SRTM / X-SAR: Products and Processing Facility*, submitted to International Geoscience and Remote Sensing Symposium, Sydney, 2001, unpublished.

3. Kiemle, S., Mikusch, E., Göhmann, M., *The Product Library – A Scalable Long-Term Storage Repository for Earth Observation Products*, DASIA 2001, Nice, 2001.