

12. GI-Fachtagung für Datenbanksysteme in  
Business, Technologie und Web (BTW 2007)  
5. bis 9. März 2007 - Aachen, Germany  
<http://www.btw2007.de/>

## Managing the Desktop Document Dataspace

Alexander Hilliger von Thile

DaimlerChrysler Research and Technology  
P.O. Box 2360, 89013 Ulm, Germany  
alexander.hilliger\_von\_thile@daimlerchrysler.com

**Abstract:** Documents such as spreadsheets are easy to create, edit, and exchange. However, their use causes a set of well known problems such as poor data quality, lack of multi user support and missing data in backend systems, e.g. for data analysis. In this paper we present our smart file prototype that extends proven concepts from DBMS to the desktop document dataspace. Firstly, we show how documents can be handled as db-objects using materialized external views. Secondly, we show how documents can be turned into a DBMS themselves.

### 1 Introduction

Many processes of an enterprise reach out to changing networks of external partners, are started spontaneously, are subject to frequent changes, and as a consequence need to be highly flexible. Due to development time and cost, for most of these processes no enterprise application and no database for data analysis exist. Furthermore, such processes are executed by non IT experts without deeper knowledge of DBMS. Thus, in these processes, documents (e.g. spreadsheets) are used for data exchange because they are easy to create, edit (even offline), and exchange. This causes problems such as poor data quality due to missing constraint checks, no up-to-date data in backend systems (e.g. for business intelligence (BI) analysis), and missing multi-user support.

Document management systems as well as recent technologies such as desktop search engines (only retrieve data without semantics) or personal data management systems (help to synchronize concurrent versions) do not tackle the underlying problem that data in documents is managed outside the control scope of a DBMS.

Recently, the data management community defined the concept of dataspace [FH+05] to better address the problem space of 'data everywhere'. Data in a dataspace (e.g. containing the documents mentioned above) can be queried and updated while the same data is accessible and updateable by existing backend systems that store their data in a DBMS, for instance. Here, the concept of dataspace allows data to be managed by existing enterprise applications (to handle core processes) as well as through documents such as spreadsheets (to support frequently changing processes as described above).

In [HvT05][HvT06] we addressed this issue theoretically by introducing the concept of '*materialized external views*' and '*smart files*'. While materialized external views handle documents as database objects, smart files turn documents conceptually into stand-alone DBMS which consist of a single file that can be exchanged by email as common documents. Both concepts aimed to extend the reach of proven DBMS

functionality to document driven processes. In this demo paper we present a prototype implementing these concepts.

## 2 Materialized External Views (MEVs)

From a DBMS perspective, a MEV can be thought of as a common materialized view that is updateable. Its content is defined by a relational expression (select-query). But, contrary to traditional materialized views, this content is materialized outside the DBMS' tablespace in a file format that is used by desktop applications (e.g. an office or CAD product). In Figure 1, creating MEV-files, working with them and reintegrating changes performed on MEVs are depicted. These steps are described next.

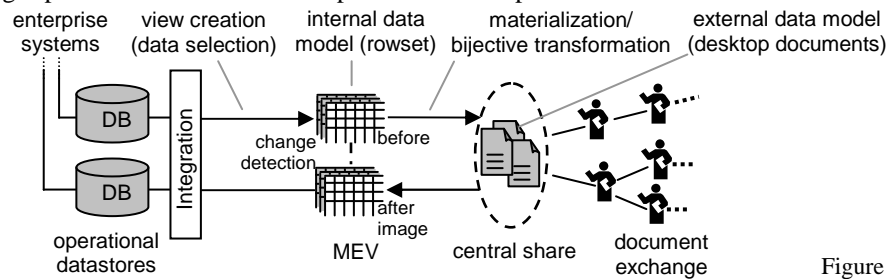


Figure 1

**Creation:** A MEV is created using a statement comparable to a common 'create view statement' that is extended by a reference to a template document (see Fig. 2). This template document contains bijective mapping definitions to map rows within the RDBMS to regions in the MEV-file and vice versa. Using the data from the (existing) base relations and the template document, the RDBMS can create the document (Fig. 3) that can be edited by the user. MEV files are under control of the DBMS, but in contrast to datalinks [MED], the DBMS not only keeps a 'link' to a complete file but also handles its contents on an attribute based granularity.

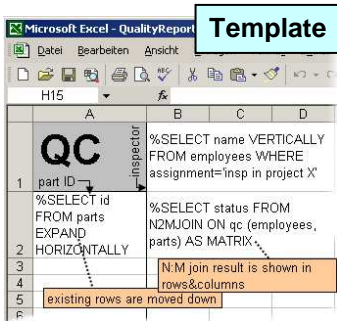


Figure 2: Template document

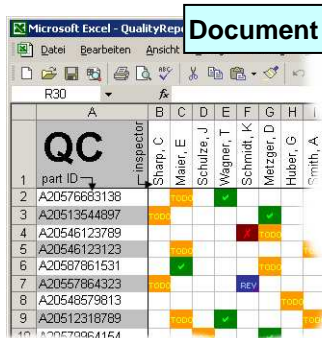


Figure 3: Resulting document

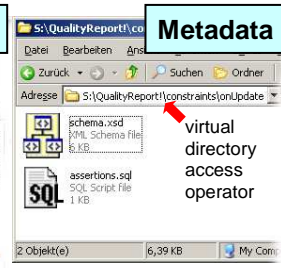


Figure 4: Attached metadata

Metadata can be attached to the MEV files (Fig. 4), e.g. to increase data quality by adding integrity constraints. Such metadata is managed in virtual directories (denoted by <filename>/) and can contain the metadata managed by the DBMS (e.g. SQL assertions) along with externally managed metadata (e.g. an XML schema document).

*Working:* Users can edit the documents using their existing, unmodified desktop applications by accessing the MEV-files provided on our managed file share.

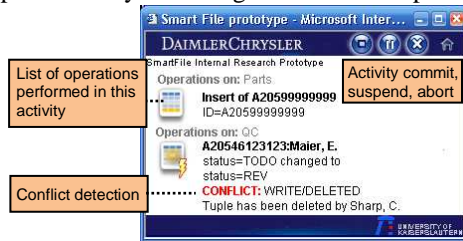


Figure 5: Journal

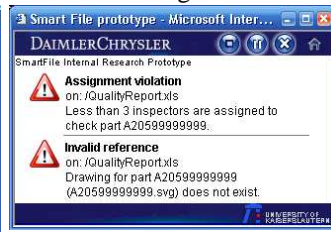


Figure 6: Constraint violations

*Reintegration:* In contrast to commonly used import/export processors, the mapped content of the view is not stored as a file. Instead, the MEV-document is a virtual file that is being managed by our prototype and that is accessible by desktop applications via a virtual (shared) drive. We use an optimistic concurrency control protocol that does not use locks. Therefore, the ‘same’ file can be edited by multiple users concurrently. To guarantee serializability of changes, we use a mechanism comparable to snapshot isolation we refer to as ‘activities’.

In Fig. 5 the user interface to start, commit, abort (rollback) and suspend activities is depicted. After an activity has been started by a user, all changes performed on files are stored within an isolated shadow copy (SC) that is invisible to other users. The SC stores all changed documents (after-images) during this activity by this user as well as the original documents (before-images). These images together with the mapping definition are used to reconstruct performed operations (insert, delete, update of rows). At the end of an activity – which is comparable to a commit in the transactional case – firstly, the change operations are reconstructed. Secondly, these changes are compared for conflicts with changes made by other users (see Fig. 5). Finally, the changes are checked for integrity (constraint-checks, Fig. 6). In contrast to DBMS, constraint violations do not roll back changes. Instead, the user has to correct the violations by editing the document again or by manually aborting the activity.

### 3 Smart Files

Downloading and exchanging MEV-files by e-mail still bears the problem that only data is being exchanged in documents. Thus, constraints cannot be checked and changes cannot be propagated to backend systems as soon as a user downloads a document. Since all benefits of a DBMS are lost if the document is being exchanged, we conceptually turned the document itself into a stand-alone DBMS by combining the document’s data with metadata and the execution logic of a DBMS.

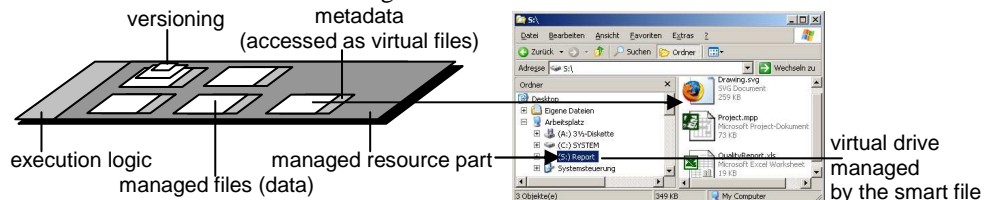


Figure 7: Concept of a smart file

The basic principle is based on a mobile DBMS consisting of a single file (Fig. 7) that can easily be exchanged by e-mail. This file is executable and contains a managed resource part where common documents and MEV-files can be stored. Metadata can be attached to these files to define integrity constraints or access rights as done in a data dictionary of a DBMS. Since this concept enhances traditional file-usage paradigms, it is called ‘smart’ file (SF) [HvT06]. A SF is a file-container, comparable to self-extracting zip-archives. It mounts its resource part as a virtual drive. However, any read/write access is managed by the SF itself, it controls who can do what, when, and where.

## 4 Related Work

In contrast to other approaches, we do not substitute document exchange, even though plenty of alternatives are available today (e.g. WfMS or groupware systems). As described in the introduction, documents are preferred by end-users for non-technical reasons that are not preserved by these alternatives (cost, development time). Today, working with documents in DBMS is handled with a mix-up of concepts from federated databases, data-integration (e.g. foreign tables in SQL/MED [MED]) and data-transformation by using import/export processors. Our approach extends the well known concept of views to bridge the gap between DBMS and documents. In the literature, plenty of work exists in the domain of view materialization (e.g. [KS+04]). However, since only data is exchanged in documents, all benefits of a DBMS are lost as soon as common documents are being exchanged. We turned documents into a stand-alone DBMS by combining the document’s data with DBMS’ execution logic and metadata. Related work can be found in the domain of electronic forms – but common documents such as CAD-files or spreadsheets can only be handled as attachments (e.g. InfoPath). A general approach is based on living [SK02] or active documents [AB+03]. However, those do not address DBMS/dataspace related issues (support for transactions, etc.).

## 5 References

- [KS+04] Karenos, K., Samaras, A, et. al.: Mobile agent-based services for view materialization. ACM SIGMOBILE, Special Issue, Volume 8, 2004.
- [AB+03] Abiteboul, S., Bonifati, A., Cobena, G., Manolescu, I., Milo, T.: Dynamic XML Documents with Distribution and Replication. SIGMOD, 2003.
- [FH+05] Franklin, M., Halevy, A., Maier, D.: From Databases to Dataspaces: A New Abstraction for Information Management, SIGMOD, Baltimore, Maryland, 2005.
- [HvT05] Hilliger von Thile, A., Melzer, I.: Smart Files: Combining the advantages of DBMS and WfMS with the simplicity and flexibility of spreadsheets. BTW. Karlsruhe, Germany, 2005.
- [HvT06] Hilliger von Thile, A.: Document Exchange Considered Useful – Extending the Reach of DBMS-Functionality to Document Driven Processes. VLDB Ph.D. Workshop, Seoul, Korea, 2006.
- [SK02] Schimkat, R.-D., Küchlin, W.: Living Documents – Micro Servers for Documents, XML-Based Data Management and Multimedia Engineering – EDBT 2002 Workshops, LNCS 2490, Prague, Czech Republic, March 2002.
- [MED] SQL/MED: see *ISO/ANSI SQL-99 standard, part 9*.