

# Integration of Workflows into a Federated Database System with *SQL/MED*<sup>1</sup>

Ralf Wagner

University of Stuttgart, Department of Computer Science  
Institute of Parallel and Distributed High-Performance Systems (IPVR)  
Breitwiesenstr. 20-22, 70565 Stuttgart, Germany  
ralf.wagner@informatik.uni-stuttgart.de

## 1 Federated Database Systems

A *federated database system* (FDBS) incorporates distributed, heterogeneous and (semi-)autonomous DBSs (see figure 1) and consists of an additional layer which provides the federation service [SL90]. This federation layer is called *federated database management system* (FDBMS) and acts as a middleware, which builds a unified view on the participating DBSs.

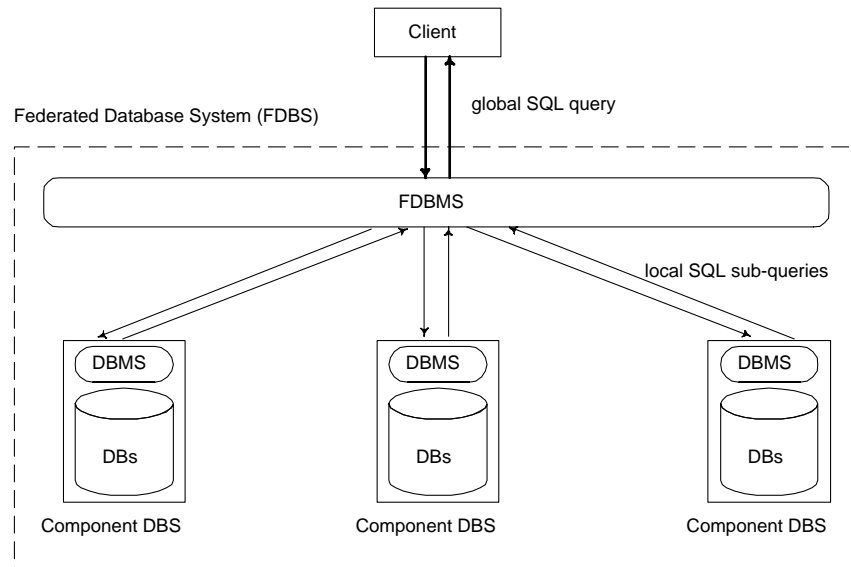


Figure 1: Architecture of an FDBS.

Users query the global conceptual schema of the FDBS. The FDBMS analyzes the query and determines the referenced component DBSs. The FDBMS then breaks down the global query, also called distributed request, into subqueries corresponding to the local conceptual schemas of the component DBSs. The component DBSs execute their subqueries and return the result to the FDBMS, which in turn composes the result of the global query and delivers it to the client.

## 2 Accessing External Functions

A widely spread type of software systems are application systems that provide access to their data only by predefined functions like CAD systems or many in-house implementations. FDBSs

<sup>1</sup>This article originated from my diploma thesis, which is available from the WWW server of the University of Stuttgart ([http://www.informatik.uni-stuttgart.de/cgi-bin/NCSTR\\_L\\_view.pl?id=DIP-1888](http://www.informatik.uni-stuttgart.de/cgi-bin/NCSTR_L_view.pl?id=DIP-1888)).

cannot access this data directly by means of a generic query language like SQL. The application programming interfaces (APIs) of these application systems must be used instead, i.e. the data-centered view is extended to include also a function-centered view [HH00].

Therefore, the given functions must be integrated into the FDBS by means of an explicit mechanism. For this purpose the *Management of External Data* (MED), a recently passed part of the SQL3 standard, can be used [ISO00]. The *SQL/MED* standard facilitates the integration of external data through employment of wrappers serving as connecting links between an FDBS and external data sources, i.e. wrappers can be used to access external functions (see figure 2(a)).

By doing so, in addition to *data federation* a so-called *function federation* can be established. Function federation can be considered as a preliminary stage of *enterprise application integration*.

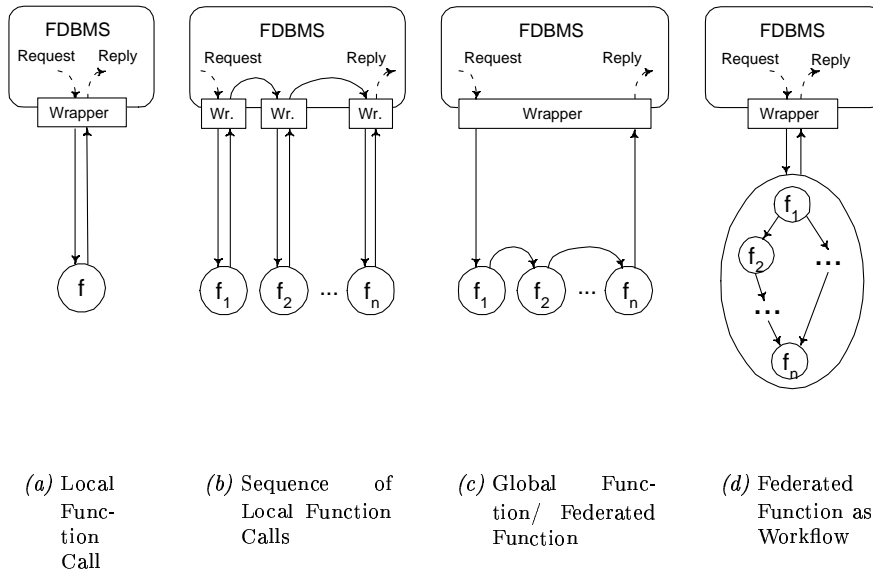


Figure 2: Functional access of application systems.

The functions of local application systems are often called in context with functions of other application systems. For example, the output data of a function of one application system is used as input data for a function of another application system (see figure 2(b)). The FDBS has to interrelate the single function calls and therefore must provide a separate wrapper for every function.

Instead of realizing such a global function directly in the FDBS it can be implemented external to it (see figure 2(c)). The benefit from that is the integration of the global function into the FDBS only once instead of every single, local function call it consists of, i.e. there is only one wrapper needed.

The global function can be regarded as a so-called *federated function* because the application systems of the accessed local functions can be distributed, heterogeneous and autonomous.

### 3 Realizing Federated Functions as Workflows

Such a federated function could be realized as a workflow (see figure 2(d)). The activities of the workflow then represent the local function calls [HH00]. The input parameters of the federated function are used as input data for the workflow representing this federated function.

The workflow logic transfers the input and output data of the local function calls from one activity to the next. Finally, the workflow logic propagates the result of the federated function

to the exit point of the workflow. The general scenario of integrating a federated function into an FDBS is shown in figure 3.

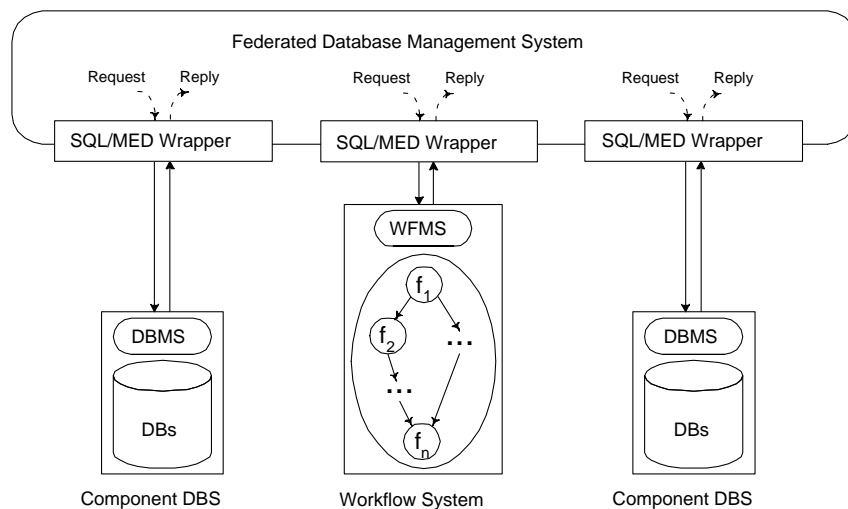


Figure 3: Integrating workflows into an FDBS via SQL/MED wrappers.

In this scenario, the federated function is realized as a workflow, which is integrated into the FDBS via an SQL/MED wrapper. There are other data sources, called component DBSs, which are also integrated into the FDBS via SQL/MED wrappers.

When the FDBS receives a global SQL query referencing a workflow, the corresponding wrapper instructs the WFMS to execute this workflow and waits for the result data. When the workflow returns the result data from its execution the wrapper has to convert and transform this data to correspond with the table format in the FDBS.

The result data of the workflow is not necessarily structured, i.e. the result data could be represented as a relational table, but it could also be represented by an XML document or a plain text file or any other data structure. The solution is to build a wrapper in accordance with the SQL/MED standard, which is responsible for a suitable conversion from the workflow result data format into a relational table format.

The input and output data structures usually are different for each workflow. Therefore, workflows need separate, customized wrappers, which are able to deal with the data structures of their respective workflows. If there are more than only a few workflows waiting for their integration into an FDBS via SQL/MED, the manual implementation of wrappers is not very economical. Hence, the next step aims at generating wrappers with a minimum of effort.

## References

- [HH00] Klaudia Hergula and Theo Härder. A Middleware Approach for Combining Heterogeneous Data Sources – Integration of Generic Query and Predefined Function Access. In *Proceedings of the First International Conference on Web Information Systems Engineering*, pages 26–33, 2000.
- [ISO00] ISO. (ISO-ANSI Working Draft) Management of External Data (SQL/MED). September 2000.
- [SL90] Amit P. Sheth and James A. Larson. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, 22(3):183–236, September 1990.