

# Metadata in a Data Grid Construction

Krzysztof Kaczmarek  
Warsaw University  
of Technology  
kaczmars@mini.pw.edu.pl

Piotr Habela  
Polish-Japanese Institute  
of Information Technology  
habela@pjwstk.edu.pl

Kazimierz Subieta  
Polish-Japanese Institute  
of Information Technology;  
Institute of Computer Science,  
Polish Academy of Sciences  
subieta@ipipan.waw.pl

## Abstract

*The paper is focused on necessary metadata definitions, playing the key role in the Grid construction. We briefly describe a realistic Grid development scenario and necessary programming and metamodel abstractions.*

## 1. Data Grid development: assumed scenario

In our research, we follow some patterns of federated database construction and extend them towards active objects and flexible (possibly – multi level) data source composition. From a practical point of view, we are skeptical about the feasibility of ad-hoc (or dynamic) integration solutions (except for perhaps very simple data structures). Thus, for any serious Grid project, a full development cycle and precise rules obliging every participant are required. We assume the following Grid construction scenario:

1. **Strategic phase** – a decision on creating a Grid is made, creates an initial integration analysis
2. **Analysis phase** – existing resources are elaborated and confronted with business requirements
3. **Design phase** – a precise definition of all schemas involved in Grid, defines roles of participants.
4. **Finalization phase** – participants sign the final agreement; all accept their roles and requirements.
5. **Implementation phase** – creation of all the necessary data adaptations described by all schemas.

The resulting specifications determine the required form of data provided both by the global service (the Grid itself), as well as each of the participants. The task of adjusting local data into the form required by the Grid is distributed between participants and the integrator. It is performed by virtual (that is, not materialized) object views [3]. The views are necessary both at the Grid level (we call them global views) for mapping multiple contrib-

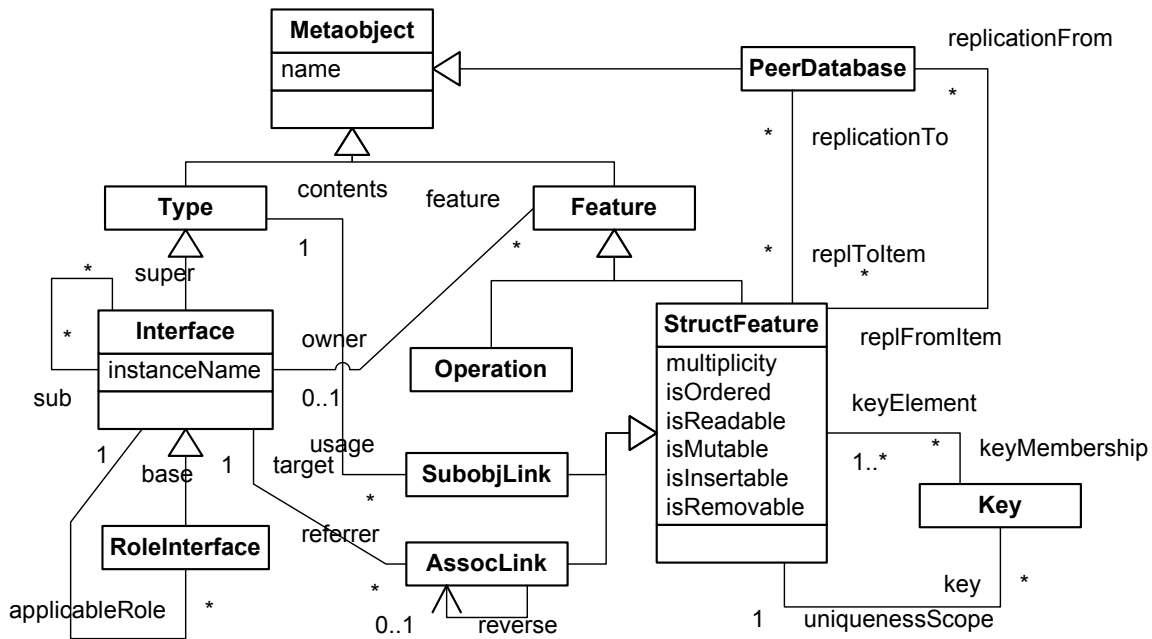
uting data sources into the globally served data, and at contributors' side (*contributory views*) to adapt local data according to the contract established by their contributory schemas. We claim that Grid nodes should be obliged to adapt their contribution as far as possible basing on the knowledge of their local resources making the final integration as simple as possible. The final note is that we follow the Stack-Based Approach [4] and assume a flexible object data model as the means of Grid integration.

The process of integration modeling deals with several levels of data and metadata, from the description of local resources up to global schemas for Grid's end-users:

- **Local schema** – the original form of data as available for node's local applications. Local schema would specify what local data are to be published (contributed) and what data remains private (local).
- **Contributory schema** – describes the data required from a given node (*contributory view*). It should make the merging of the described contribution into the global schema as straightforward as possible.
- **Global schema** – the integrated data in form as visible to the Grid clients. It may serve as a contributory schema for higher-level integration.
- **Integration schema** – specifies the transformation of contributory schemas into global schema. The mapping is realized by global virtual view definitions.

## 2. Metadata in Grid construction

In our metamodel (see the figure) we follow the above assumptions and modify popular object model to reduce secondary notions and achieve a higher level of object relativism. The notions introduced here constitute the core of the contributory schema, which is derived from a local schema to describe the data provided for the integration. We assume the full uniformity of notions between contributory schema and global schema, which helps to reduce the integration layer complexity.



**Fig. Core constructs of the contributory schema metamodel**

An important extension compared to traditional object models is the introduction of dynamic object role notion [2]. To specify objects' features we use multiplicities and changeability flags instead of collection constructors and accessibility constraints.

The means to specify single and composite uniqueness keys (playing an important role in matching data from different sources), are provided. Note that if a given interface is used in different places of database structure, for each of those places a different uniqueness constraint can be defined. Each such constraint concerning complex objects is scoped with respect to particular property declaration rather than to the *extent* of a given interface.

Operation signature specification (not shown in the figure) allows declaring arbitrarily complex structures that may result from using queries to construct the input and output parameters. Input and output data structures are treated uniformly and can take form of: a typed value (simple or complex object reference), binder (a named contents) or a constructed value (containing one or more binders). The items may be specified as multiple (and, if so, as ordered or unordered).

Another metadata important for a distributed data environment is the specification of replication paths established among participants. For each structural feature (database object or interface-hosted subobject) it is possible to indicate foreign databases to which the request of updating a copy of a given property is forwarded and others, from which it may be received.

The integration schema is realized implicitly through the global views definitions that bridge the set of contribu-

tory schemas with the global schema. They are written in high-level SBQL query language, offering the necessary simplicity and expressiveness. We believe that the benefit of introducing a graphical language for procedural constructs and objects' update behavior constituting this metadata level would be minimal.

The scope of provided metadata is limited to the technical description of data and general modeling capabilities. It is possible because Grid data sources are subjects of design-time analysis and thus people involved in Grid creation can handle the semantic details of the data. In this case no higher-level standardized data semantic description is necessary. To realize a more dynamic ("automatic") integration scenario, the amount of metadata may need to be significantly extended. The physical structure of the schema repository we are developing [1] is prepared for such extensions.

### 3. References

- [1] P. Habela, K. Subieta: Overcoming the Complexity of Object-Oriented DBMS Metadata Management. OOIS 2003: 214-225
- [2] A. Jodłowski, P. Habela, J. Płodzień, K. Subieta. Objects and Roles in the Stack-Based Approach. Proc. of DEXA Conf., Springer LNCS 2453, pp. 514-523, Aix-en-Provence, France, 2002
- [3] H. Kozankiewicz, J. Leszczyłowski, K. Subieta. *New Approach to View Updates*. Proc. of the VLDB Workshop Emerging Database Research in Eastern Europe, Berlin, Germany, 2003
- [4] K. Subieta, C. Beeri, F. Matthes, J.W. Schmidt: A Stack-Based Approach to Query Languages. East/West Database Workshop 1994: 159-180