Compositional approach for heterogeneous sources registration at a subject mediator

D.O.Briukhov, L.A.Kalinichenko, S.A.Stupnikov

Institute for Problems of Informatics RAS
{brd,leonidk,ssa}@synth.ipi.ac.ru

Abstract

The paper presents an approach for heterogeneous information sources registration at subject mediators. The information source registration is considered as the process of compositional information systems development. The method is applicable to wide class of source specification models representable in hybrid semistructured/object canonical mediator model.

1. Introduction

This paper is related to the project on building large heterogeneous digital repositories interconnected and accessible through global information infrastructures [10]. New models, theories and frameworks are to be developed in order to understand the complex interactions between various components in a globally distributed information repositories. To provide for interoperability of heterogeneous information objects, it is required to establish a global, uniform view of the underlying digital sources and services. It is assumed that specific, intermediary layer is formed by mediators providing a uniform schema and query interface to the multiple data sources to free the user from having to locate the relevant sources, query each one in isolation, and combine manually the information from the different sources.

The mediator architecture [12] deals with the problem of integration of heterogeneous information. The sources are ”heterogeneous” on many aspects: data model used, types of data, the underlying data units, behavior of objects involved, the underlying concepts, an extent to which a schema that the information may conform can be made rigid in advance. Examples of ”semi-structured” information include those found in XML documents, repositories used in the bio-molecular data, Web sites, etc.

In this particular project *subject mediators* are emphasized that support representation and access to various subject domains. Mediators should provide modeling facilities and methods for conversion of unorganized, nonsystematic population of sources registered by different source providers into a well-structured set of sources supported by the integrated uniform specifications. On the mediator’s layer the users are provided with the metainformation uniformly characterizing subject content of the underlying sources and the canonical information model

making possible to query such sources and 'compute' the response. This model is needed to express the structure and semantics of the integrated data as well as the available services. Each mediator supports the process of registration of sources providing the uniform ontological knowledge and metainformation to support discovery and compositions of existing sources. This process is considered as a semi-automatic. It is expected that source providers (the original capital investors) will be interested in registering their sources in a common pool in mediators to optimize the investments.

The mediator's metainformation is intended to be shared by information consumers, source providers and subject mediators. The paper shows the broad range of information models that need to be uniformly represented in mediators. The canonical information model intended for uniform representation of heterogeneous metainformation is introduced. An approach for equivalent representation of different kinds of information models in the canonical one is considered.

Creation of the metainformation on the interoperation level is specifically emphasized. The metainformation registry system uses the canonical model constructs to link diverse contexts and representation of heterogeneous metadata among themselves. Acquisition and integration of metadata defining the information sources' content and capabilities in each domain are the basic functions of mediators. Metainformation assists in selection of sources relevant to a query, generation and optimization of queries against the source [5].

The paper starts with an analysis of diversity of information that should be presented at the mediator's level and with brief characterization of the canonical model that is required for the mediator. An approach for uniform representation of heterogeneous sources in the canonical paradigm is discussed. The process of heterogeneous information source registration at subject mediators with local as view (LAV) organization is presented. A tool supporting the registration process is briefly discussed. Formal facilities required for justification of refinement relationship during the process of registration are also presented.

2. Heterogeneous sources diversity

The broad range of information modeling facilities should be considered for their representation at the mediator's level, including those for textual and multimedia information, heterogeneous databases, ontological information, unstructured and semistructured information, heterogeneous object components, e.g.:

- Semistructured data modeling facilities emerging to model the Web itself, structure of Web sites, internal structure of Web pages, and contents of Web sites;
- Metadata expressible in metamodels (such as the World Wide Web Consortium's Resource Description Framework (RDF) designed for exchanging machine-understandable metadata describing Web resources);
- Knowledge representation models expressible in well-known notations including the language for knowledge communication based on the predicate calculus semantics (KIF), a model for maintaining ontologies portably in a form that is compatible with

multiple representation languages (Ontolingua), a common knowledge model of various knowledge bases (OKBC);

• Heterogeneous object component modeling facilities including interface specifications providing for technical interoperability (IDL), and definitions providing more semantics for component-based development (BOF, CDL);

• Object models for the Web representing a document as a hierarchy of objects which are derived (by parsing) from a source representation of the document (HTML or XML) – Document Object Model (DOM);

• Object and heterogeneous database models characterized by the basic standards for object modeling (ODMG ODL), object-relational modeling (SQL:1999), as well as by the heterogeneous multidatabase modeling.

To homogenize such variety of models uniformly representing them in one paradigm a specific approach has been developed providing for mapping of various data models and metainformation into the canonical one using principle of data model refinement [1, 6].

Main idea of the approach consists in creation of extensions of the canonical model core for each data/knowledge model that may be used for a digital source representation. These extensions should be formed so that these models become their justifiable refinements. Satisfying this condition guarantees preservation of information and operations while mapping various models and respective metadata into the common paradigm. This approach is the basic one for the uniform representation of different digital sources [8]. Fig. 1 shows canonical model core extensions to be formed for various information models considered above. Canonical model is formed as a union of the core with all extensions obtained.

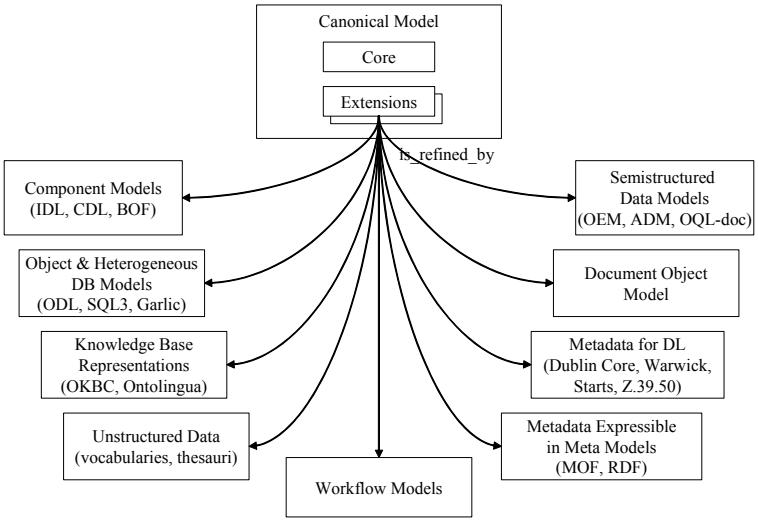

Fig. 1. Canonical model core extensions

3. Canonical model core for the mediator's environment

We base the mediator's canonical model core on the SYNTHESIS language [6] that has been elaborated for semantic interoperation and component based information systems development in the wide range of pre-existing heterogeneous information sources. The language possesses hybrid capabilities providing for integration of structured as well as semi-structured data models [8]. A set of the canonical model facilities used for the uniform representation of the information sources includes the following:

- Frame representation facilities. Frames are treated as a special kind of abstract self-defined values introduced mostly for description of concepts, terminological and weakly-structured information. In particular, information source metainformation (schema) is represented using the frame language. Frame representation facilities provide for expressing of arbitrary semantic associations of frames, unstructured, textual and temporal associations. All specifications in the canonical model have a form of frames that become a part of the metainformation base;
- Unifying type system. A universal constructor of arbitrary abstract data types as well as a comprehensive collection of the built-in types are included into a type system. A type specialization (subtyping) relationship is a part of the model. Types are values themselves. Metatypes provide for classification of the type hierarchy. Type expressions are introduced for type compositions that are required to type the results of queries and of heterogeneous component compositions;
- Class representation. Classes provide for representing of sets of homogeneous entities of an application domain. Class hierarchies and type inheritance mechanisms make possible to define the generalization / specialization relationships. Class instances (objects) have specific types. Metaclasses provide for introducing different classification relationships orthogonal to the class generalization relationship;
- Multiactivity (workflow) representation. These are used for the specification and implementation of interconnected and interdependent application activities, for the specification of declarative assertions and concurrent megaprograms over the information sources. These facilities provide for specification of concurrent and asynchronous behaviour of application systems and of interoperable source environments as of dynamic discrete events systems;
- Facilities for the logical formulae expressions. A multisorted object calculus (typed first-order language) is used for querying the integrated set of digital sources as well as for specification of constraints and behaviour.

Schematically basic entities of the canonical model and their relationships are shown on Fig. 2.

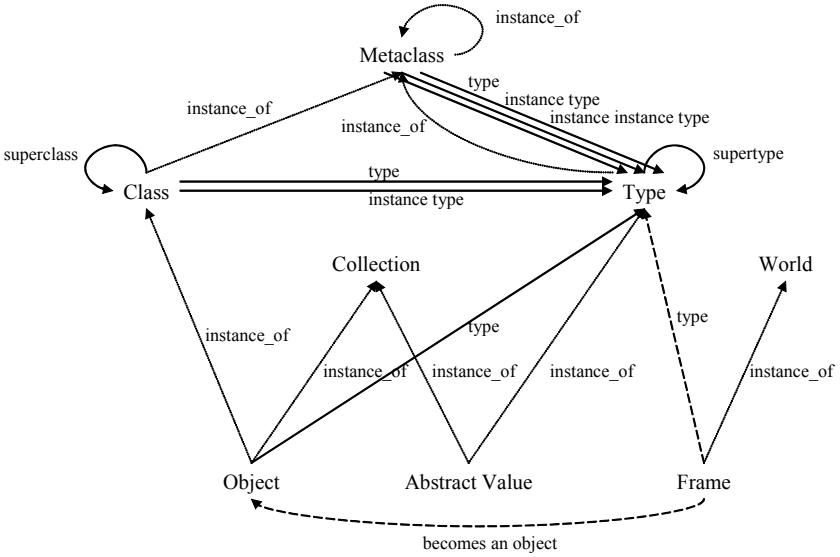

Fig. 2 Canonical model entities

4. Registration of Information Sources at a Subject Mediator

According to the LAV mediation, the registered sources schemas are considered as materialized views above virtual classes of mediator. In the specific registration method reported [4] the materialized views mentioned are designed applying compositional development method (source schema is treated as a specification of requirements and class schemas of the mediator are treated as component specifications). This approach is intended to cope with a dynamic, possibly incomplete set of sources. Sources may change their exported schemas, become unavailable from time to time. To disseminate the information sources, their providers should register them at a respective subject mediator. Such registrations can be done concurrently and at any time. Specific methods and tools supporting process of information sources registration have been developed to make mediators scalable with respect to a number of sources involved.

During the registration a local source class is modeled as a set of instances (objects) of the class instance type, and a description of the source in terms of the mediator schema specifies the constraints on the class instances to be admissible for the subject mediator. The process of registration includes the ontologically-based reconciliation of the application contexts of the registered sources and that of the mediator, identification of relevant classes of the mediator schema, constructing of the most common reducts for the mediator type specifications and respective types of the sources, constructing of views, specifying source class constraints in terms of the mediator classes.

The information source registration is considered as the process of compositional information systems development [3]. Local source metainformation definitions are

treated as specifications of requirements and classes of the federated level with the related metainformation - as specifications of pre-existing components. To get local classes definitions as views above the federated level with constraints given in the terms of the federated classes, the facilities of the modified compositional information systems development method and tool [3] are applied.

4.1 Fundamentals of the compositional IS development method

The main distinguishing feature of the compositional IS development method is a creation of compositions of component specification fragments refining specifications of requirements. Widely used component-based development methods (e.g., JavaBeans) construct aggregates of components differently - just linking ports of components with each other or considering their interactions on the contractual basis.

Refining specifications obtained during the compositional development, according to the refinement theory, can be used anywhere instead of the refined specifications of requirements without noticing such substitutions by the users. The refinement methods allow to justify the fact of a refinement formally to guarantee the adequacy of the specifications obtained to that of the required.

The compositional development is a process of systematic manipulation and transformation of specifications. Type specifications of the mediator canonical model (SYNTHESIS) are chosen as the basic units for such manipulation. The manipulations required include decomposition of type specifications into consistent fragments, identification of reusable fragments (patterns of reuse), composition of identified fragments into specifications concretizing the requirements, justification of reusability and substitutability of the results of such transformations instead of the specifications of requirements. The compositional specification calculus [9] intentionally designed for such manipulations uses the following concepts and operations.

A signature $\sum_T$ of a type specification $T = <V_T, O_T, I_T>$ includes a set of operation symbols O_T indicating operations argument and result types and a set of predicate symbols I_T (for the type invariants) indicating predicate argument types. Conjunction of all invariants in I_T constitutes the type invariant. We model an extension V_T of each type T (a carrier of the type) by a set of proxies representing respective instances of the type.

Definition 1. Type reduct. A signature reduct R_T of a type T is defined as a subsignature $\sum'_T$ of type signature $\sum_T$ that includes a carrier V_T, a set of symbols of operations $O'_T \subseteq O_T$, a set of symbols of invariants $I'_T \subseteq I_T$.

Definition 2. Type U is a **refinement** of type T iff

- there exists a one-to-one correspondence *Ops* between O_T and O_U;
- there exists an abstraction function *Abs*, that maps each admissible state of T into the respective state of U;
- invariant of type U implies (under *Abs*) invariant of type T;
- for every operation o in T the operation $Ops(o) = o'$ in U is a refinement of o. To establish an operation refinement it is required that operation precondition

pre(o) should imply the precondition *pre(o')* and operation postcondition *post(o')* should imply postcondition *post(o)*.

Based on the notions of reduct and type refinement, a measure of common information between types can be established.

Definition 3. A **common reduct** for types *T1*, *T2* is such reduct R_{T1} of *T1* that there exists a reduct R_{T2} of *T2* such that R_{T2} is a refinement of R_{T1}. Further we refer to R_{T2} as to a conjugate of the common reduct.

Definition 4. A **most common reduct** $R_{MC}(T1,T2)$ for types *T1*, *T2* is a reduct R_{T1} of *T1* such that there exists a reduct R_{T2} of *T2* that refines R_{T1} and there can be no other reduct R'_{T1} such that $R_{MC}(T1,T2)$ is a reduct of R'_{T1}, R'_{T1} is not equal to $R_{MC}(T1,T2)$ and there exists a reduct R'_{T2} of *T2* that refines R'_{T1}.

Reducts provide for type specification decompositions thus creating a basis for their further compositions. Type composition operations can be used to infer new types from the existing ones. We introduce here definition of only one such operation - *join*.

Definition 5. Type **join** operation. An operation *T1* | *T2* produces type *T* as a 'join' of specifications of the operand types. Generally *T* includes a merge of specifications of *T1* and *T2*. Common elements of specifications of *T1* and *T2* are included into the merge (resulting type) only once. The common elements are determined by another merge - the merge of conjugates of two most common reducts of types *T1* and *T2*: $R_{MC}(T1,T2)$ and $R_{MC}(T2,T1)$. The merge of two conjugates includes union of sets of their operation specifications. If in the union we get a pair of operations that are in a refinement order then only one of them, the more refined one (belonging to the conjugate of the most common reduct) is included into the merge. Invariants created in the resulting type are formed by conjuncting invariants taken from the original types.

A type *T* is placed in the type hierarchy as an immediate subtype of the join operand types and a direct supertype of all the common direct subtypes of the join argument types. If *T2* (*T1*) is a subtype of *T1* (*T2*) then *T2* (*T1*) is a result of a join operation.

Operations of the compositional calculus form a *type lattice* [9] on the basis of a subtype relation (as a partial order).

4.2 Process of an information source registration

The process of an information source registration at the subject mediator includes the following steps:

1. relevant federated classes identification
   For each *source* class find federated classes that ontologically can be involved to form the source class extent. To the source class several federated classes may be relevant covering with their instance types different reducts of an instance type of the source class. On the other hand, several source classes may correspond to one federated class.

2. most common reducts construction
   For an instance type of each identified *federated* class construct most common reducts for instance type of this federated class and source class instance type to concretize (partially) such federated instance type. Most common reduct may include also additional attributes corresponding to those federated type attributes that can be derived from the source type instances to support them. In this process for each attribute type of the common reduct a concretizing type, concretizing function or their combination should be constructed (this step should be recursively applied).
3. partial source view construction
   For each relevant federated class construct a partial source view expressing constraints in terms of the federated class that should be satisfied by values of respective most common reducts of source class instances. Thus partial views over all relevant federated classes will be obtained.
4. partial views composition
   Construct compositions of the source type most common reducts obtained for instance types of all federated classes involved. Construct a source view as a composition of partial views obtained above. This is an expression of a materialized view of an information source in terms of federated classes. An instance type of this view is determined by the most common reducts composition constructed above.

5. Formal justification of heterogeneous sources registration at a subject mediator

In the subject mediation project formal facilities are required for various transformations applied. Data model mapping, query rewriting, compositions of ontologies are just few examples of those. Here a brief overview of formal facilities based on a notion of refinement and intended for justifications of such mappings and transformations is presented. To be specific, these facilities are aimed here at the most common reducts construction and partial views composition during the source registration at the mediator.

Most common reducts are to be constructed for all pairs of relevant federated class instance types and source class instance types. A reduct of a source class instance type is to be a refinement of the respective reduct of a federated class instance type.

Source view is constructed as a composition of partial views over relevant federated classes. An instance type of the view is constructed as a composition of concretizing reducts of the source classes instance types. Operations of type composition (meet and join) are also based on the notion of refinement.

The notion of refinement is to be formalized and the fact of refinement is to be formally proven. For these purposes the Abstract Machine Notation (AMN) and B-technology are applied [1]. AMN is a formal specification language based on first order predicate logic and set theory formalizing the notion of refinement. Data structures in AMN are expressed by means of set theory and receive a precise mathematical meaning. Operations over data are specified as generalized substitutions

that generalize Dijkstra guarded commands. A modularization concept of the AMN is an abstract machine allowing to organize large specifications as independent fragments having well-defined interfaces. B-technology is a pack of methods and tools aimed at proving the fact of refinement semi-automatically. As a tool supporting the AMN, B-Toolkit [2] by B-Core has been chosen.

During registration all the specifications of the instance types – federated classes instance types and source classes instance types – are represented in the canonical information model. To justify a refinement relation between canonical specifications formally, a method of mapping the canonical model into the AMN has been developed [11].

General idea of mapping of canonical model types to the AMN consists in extensional interpretation of types. An extent of each type (a set of admissible instances of a type) of a canonical specification is represented by finite constant set. These sets are related in accordance with the subtype relationship so that an extent of a subtype becomes a subset of an extent of a supertype. Thus a primary model of the type hierarchy declared in the canonical model is established. Extensional constants provide a natural basis to model a type specification by a collection of functions of AMN. Each function corresponds to a state attribute of the type. A domain of a function is an extent of the type and a range of the function is an extent of the attribute value. Functional attributes (operations) of canonical model types may have dual representation in their mapping to AMN: 1) they are represented as operations of abstract machines; 2) they may additionally be represented as variables of abstract machines. These variables are typed in accordance with a signature of the canonical model operations. For each variable a predicate showing how input arguments of the operation are related to output parameters in accordance with a predicative specification of the canonical model operation is presented in an abstract machine. Additional representation of operations as functional variables is introduced for operations having no side effects to model operation calls within type invariants and recursive operation calls.

Another aspect of the mapping is a representation of the compositional structure of the canonical model type definitions. This structure can be interpreted by a directed compositional graph (DCG) determined by set N of nodes corresponding to type specifications and set E of edges. An edge $e=<T_1,T_2>$ where T_1 and T_2 are nodes, belongs to E if there is an access in specifications of attributes of T_1 to attributes of T_2. The intention is to provide a representation of DCG by compositional structure of separate abstract machines. Unfortunately modularization features of AMN can not cope with some properties of DCG, i.e. cycles. This can be resolved by transforming DCG into an acyclic graph and by mapping of each node of the resulting graph into a distinct abstract machine.

To prove that the mapping of the canonical mediator model into the AMN is correct a method for justification of the mapping Θ of specification languages has been developed. Θ is based on the combined denotational and axiomatic semantics of the languages.

Current work for formal justification of heterogeneous sources registration at the mediator consists in development of a tool providing for automatic mapping of canonical specifications into AMN specifications.

6. A tool for registration of heterogeneous sources at the mediator

Figure 3 shows a general structure of the source registration tool.

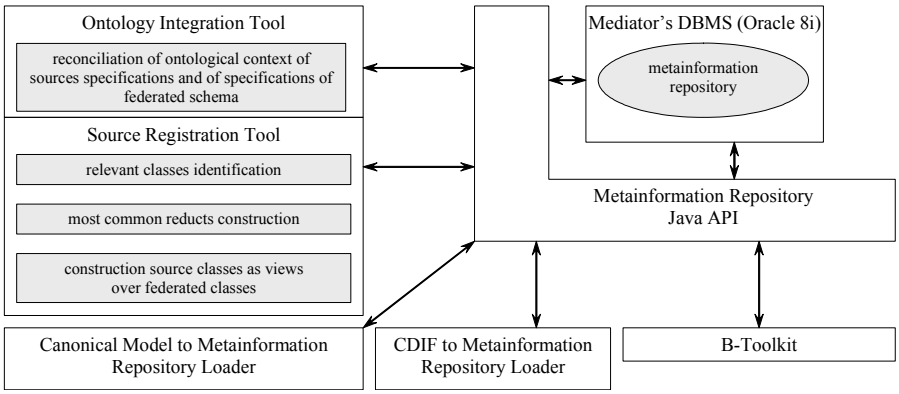

Fig. 3 General structure of the sources registration tool

Source registration tool has been developed [4] on the basis of the SYNTHESIS compositional design method prototype [3]. It has been developed using Java 2 under Windows environment.

The Oracle 8i is used to support the metainformation repository. To access the metainformation repository a specific Metainformation Repository Java API is used.

PLATINUM Paradigm Plus has been chosen to support the Requirement Planning and Analysis phases. UML notation can be used to represent specifications of requirements and specifications of sources in a graphical form.

Using the CDIF representation taken out of the Paradigm Plus, the specifications obtained on the early stages of development can be loaded into the metainformation repository.

For formal modeling the B Abstract Machine Notation is used that together with B-Toolkit [2] provides for type specification consistency check, adequacy of component specifications, establishment of a refinement condition, generating and proving respective proof obligations.

7. Conclusion

The paper presents an approach for heterogeneous information sources registration at subject mediators. The information source registration is considered as the process of compositional information systems development. The method is applicable to wide class of source specification models representable in hybrid semistructured/object canonical mediator model. Complete type specifications are basic for the method. Ontological specifications are used for identification of mediator classes semantically relevant to a source class. Maximal subset of source information relevant to the mediator classes is identified (use of the most common reducts leads to identification

of maximal commonality between a source and federated level class specifications). Concretizing types are defined so that federated classes instance types are refined by the source instance type. This is natural direction supporting a query plan refining a mediator query in terms of a specific source. Such refining direction is in contrast to conventional compositional development where specification of requirements is to be refined by specifications of components. Formal facilities required for justification of refinement relationship during the process of registration are also discussed.

References

[1] J. -R. Abrial. The B-Book. Cambridge University Press, 1996

[2] B-Toolkit User's Manual - Release 3.4. Copyright B-Core (UK) Ltd., 1997

[3] Briukhov D.O., Kalinichenko L.A. Component-Based Information Systems Development Tool Supporting the SYNTHESIS Design Method. In Proc. of the East European Conference on Advances in Databases and Information Systems, Poland, Springer, LNCS No.1475, 1998, pp. 305-327

[4] Briukhov D.O., Kalinichenko L.A., Skvortsov N.A. Information sources registration at a subject mediator as compositional development. In Proceedings of the Fifth East European Conference on Advances in Databases and Information Systems (ADBIS'01), Springer, LNCS No.1251, 2001, pp. 70-83

[5] O.Duschka, M. Genesereth, A. Levy. Recursive query plans for data integration. Journal of Logic Programming, special issue on Logic Based Heterogeneous Information Systems, 43(1): 49-73, 2000

[6] Kalinichenko L.A. SYNTHESIS: the language for description, design and programming of the heterogeneous interoperable information resource environment. Institute for Problems of Informatics, Russian Academy of Sciences, Moscow, 1993, 103 p.

[7] Kalinichenko L.A. Method for data models integration in the common paradigm. In Proceedings of the First East European Conference 'Advances in Databases and Information Systems', St. Petersburg, September 1997

[8] Kalinichenko L.A., Integration of heterogeneous semistructured data models in the canonical one. In Proceedings of the First Russian Conference on "Digital Libraries: Advanced Methods and Technologies, Digital Collections", Saint-Petersburg, October 1999

[9] L. A. Kalinichenko. Compositional Specification Calculus for Information Systems Development. In Proc. of the East-Europeant Conference on Advances in Databases and Information Systems (ADBIS'99)}, Maribor, Slovenia, September 1999, Springer Verlag, LNCS, 1999

[10] Kalinichenko L.A., Briukhov D.O., Skvortsov N.A., Zakharov V.N. Infrastructure of the subject mediating environment aiming at semantic interoperability of heterogeneous digital library collections. In Proceedings of the Second RCDL, October 2000, Protvino

[11] Stupnikov S.A. Mapping of Specification Canonical Model to Formal Notation for Refining Specifications Modelling. In Proceedings of the XXIV Conference of Young Scientists, Faculty of Mechanics and Mathematics, Moscow State University, April 8-13, 2002, Moscow

[12] G. Wiederhold. Mediators in the Architecture of Future Information Systems. IEEE Computer, 1992