Ersch. in: On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops ; Vilamoura, Portugal, November 25-30, 2007 ; vol. 1 / Robert Meersman ... (eds.). - Berlin : Springer, 2007. - S. 5-6. - (Lecture notes in computer science ; 4805). - ISBN 978-3-540-76887-6

Generic Proxies—Supporting Data Integration Inside the Database

Andrei Vancea, Michael Grossniklaus, and Moira C. Norrie

Institute for Information Systems, ETH Zurich
CH-8092 Zurich, Switzerland
{vancea,grossniklaus,norrie}@inf.ethz.ch

Abstract. Existing approaches to data integration generally propose building a layer on top of database systems to perform the necessary data transformations and manage data consistency. We show how support for the integration of heterogeneous data sources can instead be built into a database system through the introduction of a generic proxy concept.

Over the last two decades a great deal of research in the database and information systems communities has addressed the challenges of data integration. Generally, the problem addressed is how to combine data from different sources to provide a unified user view [1]. Various approaches have been proposed depending on the purpose of the integration and the nature of the data sources, but two broad categories of data integration systems that have received a lot of attention in recent years are *mediator* [2] and *data warehousing* [3] systems. These systems tend to have a common architectural approach in that integration is achieved by building extra layers on top of the database systems. We believe that adding internal support for data integration in a database system can have positive effects in the development of data integration systems.

In our approach, the integration of external information sources is done using a *generic proxy*. A generic proxy consists of two parts: the *proxy object* and *the proxy process*. The proxy object represents the database view of the external data source. The data from the external source is cached locally, similar to the *data warehouse* approach. Queries can be executed locally without any communication to the external source. The synchronisation between the database view of the information source and the external information source is done automatically by the database management system in a transparent way. We have defined a proxy programming interface that allows the user to specify how a proxy object interacts with an external source. The user has to write different implementations for different types of external sources. The proxy processes are created from particular implementations of the proxy interface.

When a user wants to create a new proxy object, they must specify the name of the proxy and also the list of arguments that are needed in order to initialize the generic proxy. First, a new proxy object is created and stored in the database. Afterwards, the proxy object must be associated with an existing or newly created proxy process. This association is performed using a chain of responsibility approach. All of the existing proxy processes pertaining to the current proxy

type are asked to accept the newly created proxy object using the *accept* call. The proxy object is associated with the first process that accepts it. If no such process is found, a new one is created and associated with the proxy object. The association between the proxy object and the proxy process cannot be changed at a later time.

A proxy process must handle the bi-directional communication between the database and the external source. When the proxy object is changed, the database system, using the proxy process, sends the modifications to the information source. At the same time, when the external information source is changed the database system is notified by the proxy process. Having a running proxy process for each proxy object is clearly not a feasible solution. We therefore chose to map more than one proxy object to a single proxy process. By using the proxy programming interface, the user can specify how the mapping of proxy objects to proxy processes is done for particular types of proxies.

We maintain a FIFO list that contains the proxy objects that are scheduled for synchronisation with their external information sources. A proxy object is added to this list if the value of one of its attribute is modified or as a result of the modification of the external information source. The proxy objects are extracted, one by one, from the list and are synchronised with the external sources. During the synchronisation process, a new object is created (*remoteObject*) by reading the data directly from the external source. The values of the two objects (the proxy object and *remoteObject*) are then merged together, resulting a new object (*mergedObject*). Potential conflicts are also solved during the merging process. The values of *mergedObject* are then sent to the information used, using the proxy process. The proxy object is replaced with *mergedObject*.

By using the generic proxy mechanism, the synchronisation between the external information sources and the database system is done automatically when the information source is changed or when the value of its proxy object is modified. The system does not guarantee that the client will work with the latest versions of the information sources, but the synchronisation is usually done within a reasonable amount of time. We have implemented generic proxies in an object data management framework based on the db4o object storage system [4].

References

1. Lenzerini, M.: Data Integration: A theoretical Perspective. In: Proceedings of ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Madison, WI, USA, pp. 233–246. ACM Press, New York (2002)
2. Wiederhold, G.: Mediators in the Architecture of Future Information Systems. In: Huhns, M.N., Singh, M.P. (eds.) Readings in Agents, Morgan Kaufmann, San Francisco (1997)
3. Widom, J.: Research Problems in Data Warehousing. In: Proceedings of International Conference on Information and Knowledge Management, Baltimore, MD, USA (1995)
4. Paterson, J., Edlich, S., Hörning, H., Hörning, R.: The Definitive Guide to db4o. Apress (2006)