Interoperability and Object Identity

Frank Eliassen
Department of Computer Science
University of Tromsø
N-9000 Tromsø

Randi Karlsen
Department of Computer Science
University of Tromsø
N-9000 Tromsø

Abstract

Data model transparency can be achieved by providing a canonical language format for the definition and seamless manipulation of multiple autonomous information bases. In this paper we assume a canonical data and computational model combining the functional and object-oriented paradigms.

We investigate the concept of identity as a property of an object and the various ways this property is supported in existing databases, in relation to the object-oriented canonical data model.

The canonical data model is the tool for combining and integrating preexisting syntactical homogeneous, but semantical heterogeneous, data types into generalized unifying data types. We identify requirements for object identity in federated systems, and discuss problems of object identity and semantical object replication arising from this new abstraction level. We argue that a strong notion of identity at the federated level can only be achieved by weakening strict autonomy requirements of the component information bases. Finally we discuss various solutions to this problem that differ in their requirements with respect to giving up autonomy.

1 Introduction

Data model transparency hides the differences in query languages and formats for the user when accessing heterogeneous information bases. This way, uniform access to heterogeneous information bases can be provided.

A requirement for providing data model transparency is the adoption of a canonical data model. The extensible nature of object-oriented data models and their good support of the principle of data abstraction, makes their role as the basis for a canonical data model quite natural [CoLy88]. The object-oriented paradigm supports data abstraction through the technique of encapsulation. The major benefit of encapsulation is its ability to overcome heterogeneity through the provision of abstract data types (ADTs) and associated operations. ADTs are essentially descriptions of potential objects in a system, and extends the principle of data abstraction by separating the specification of a data abstraction from its implementation. To use an ADT, it is only necessary to know the specification. There is absolutely no need to know about the implementation. This makes it feasable to write object-oriented interfaces to a wide variety of existing information bases.

Considering entity types of data models as ADTs, provides a syntactical homogeneous view of information bases. However, although corresponding ADTs in different information bases are specified using the same language, they may still be semantically heterogeneous in the traditional sense of differing data structures or measurement units. We do not address the reconciliation of such issues in this paper beyond arguing that appropriate semantical mappings can be specified using the canonical data manipulation and programming language facility of the global level.

Frequently the canonical data model for defining an object oriented view has been combined with a functional data model and functional programming language. This has also been our approach [ElKa91]. We model a database as a computational object encapsulating a set of passive (complex) data objects. Object methods and attributes are syntactically treated as functions and hence can be combined with built-in function symbols. Through the support of higher order functions, it also becomes possible to cope with the kind of schematic discrepancies described in [KrLK91] where one databases's data correspond to metadata in others.

Identity is that property of an object which distinguishes it from all others [KhCo86]. Unlike the relational data model, a prominent feature of most object oriented models is the support for a rather strong notion of object identity. Being object-oriented, one might expect the canonical data model to support a strong notion of object identity as well. Indeed, for some federated level functions like transaction management, a strong notion of identity *is* required. A conflicting requirement, however, is that the canonical data model must provide this notion of identity based on component information bases supporting identity varying from the strongest forms of identity to the weaker forms only. The discussion of this issue is the focal point of this paper.

In the next Section we summarizes the various ways object identity is supported in existing information systems. Section 3 discusses requirements to object identity at the federated level. In Section 4 we present a canonical object model focusing on the representation of global object identity. Section 5 discusses different approaches to how a federated system may strengthen a given notion of object identity in relation to requirements to autonomy of component information bases. In Section 6 we extend the object model with the notion of semantical replication, and discuss how semantical replica can automatically be identified by the federated system. The last Section summarizes.

2 Identity in data models

Information bases using different data models may support different notions of object identity. For our purpose we are concerned about identity as it is made available to the federatated information system seen as a client of the different component information bases. We have identified three different support levels of object identity that may be provided to the federated system by component information bases. This is either value based identity, session object identifiers (oids), or immutable oids. We do not consider the internal representation of identity in each component information base.

An immutable oid gives the object a unique identification which is independent of object value and location, and lasts for the lifetime of the object. Immutable oids provide a strong notion of identity. A strong notion of identity at the federated level requires a secure object identification. An object identification is *secure* if it always denotes the same object independent of how the object value may change. It follows that an immutable oid is secure.

Value based identity and session oid allow changable identification of objects. The identity of an object using session oids generally changes from one session to another. In systems supporting value based identity, the identity of an object is changed every time the value of the identifying key attribute changes.

By a session we shall mean an execution of a group of operations on a component information base such that during the session, the identification of the set of accessed objects is secure. Thus a session can be characterized as the unit of secure identification in a component information system. For databases supporting value based identity only, a session will normally correspond to a (local) transaction, while for other (e.g. object-oriented) databases, a session can correspond to a program execution that may encompass several transactions (see below).

For the client of an information base, session oids, and value based identity, are *insecure*, since, at least at the start of the session, objects must be retrieved from the information base by selecting on attribute values. Hence a client can generally not tell whether a given query expression will retrieve the same object when applied in two different sessions.

Most database systems use identifier keys to distinguish persistent objects, mixing data value and identity. Concepts similar to both immutable and session oid are used in many system, as e.g. in the object oriented DBMS O_2 [O2Te91]. During program execution, the O_2 DBMS engine provides to its clients session oids (O_2 handles) that may be assigned to object variables. These handles are secure object identifiers within the scope of a single program execution. A concept corresponding to immutable oid is also supported in O_2, by allowing for named objects in the schemata[1]. These objects are the root of persistency. A federated system can easily maintain a mapping between global immutable oids and named objects in the local schema. Similarly, the roots of persistency in a relational database correspond to a set of named relations. However, relational databases do not support a concept corresponding to session oids.

3 The problem

The canonical object model of the federated system is the tool for defining new (and complex) object types at the federated level. These object types and corresponding objects, we refer to as *global*. Their definition (and population) is directly or transitively based on existing, syntactically homogeneous, but possibly semantically heterogeneous, object types exported by different information bases. An issue is whether global objects have a persistent global representation, or are "imaginary" in the sense of [AbBo91]. By persistent we mean an instantiation of an object representation that survives sessions, while instantiations of "imaginary" global object representations only last a session. Clearly, global object types that have attributes that can not be derived from attributes from exported types, must be stored globally and thus require a persistent global object representation.

However, requirements for a persistent global representation of at least some aspects of global objects (like their identity), also originate from functions at the federated level. In particular, our analysis shows that federated transaction mangement in many cases requires a notion of identity that sometimes can be difficult to support by component information bases. Transaction models for federations in which component systems have strong autonomy requirements, usually assume non-isolated transactions with recovery based on compensation [ELLR90]. Compensation is modelled as a separate normal transaction that semantically undoes the effect of some previous (sub)transaction. This must be done by identifying the data (or objects) on which the compensating actions is to be invoked. Obviously, the identity of the involved objects should not be allowed to change between the first (sub)transaction and its associated compensating transaction. Otherwise the compensation might fail since the object identification is not secure.

For component information systems where a session corresponds to a local transaction, the primary subtransaction and its compensating transaction must be mapped to different sessions. To solve the resulting identification problem, an additional mechanism

[1] Object names are immutable at least as long as no changes to the local schema are made

that preserves the identity of the involved objects between the two related sessions, is required. For systems that allow several transactions within a session, the object identity of the involved objects will be preserved between the primary sub-transaction and its compensating transaction if the two transactions are executed within the same session. Otherwise an additional mechanism as above, is also required in this case.

Note that the above requirements for identity is in fact independent of the kind of data model used at the federated level. However, the object-oriented data model clearly has an advantage as compared to other data models since object identity is an inherent notion of the object-oriented paradigm. Anyhow, the problem that arises is that the notion of identity required at the global level, may be stronger than that offered by the component information bases to their clients (including the federated information system itself).

Following [KhCo86], a surrogate-based implementation scheme is needed to support a strong notion of identity. This requires a persistent global representation of the identity of global objects that survives sessions. The global representation must be instantiated at object type definition time by a query towards the appropriate component information bases. At a logical level, this representation must for each global object include a mapping from the object's global identity to the local identity of the corresponding object in a component information base.

A global representation of the identity of global objects where the local identities are based on session oids and value based identity, needs some additional mechanisms to support a strong notion of identity. This is because the local identity of objects may change from one session to the other without the global level knowing it.

Hence there is sometimes a need to strengthen the notion of identity provided by some component information bases such that the identity of objects is preserved across sessions. In the following we discuss this issue in light of requirements for autonomy. We argue that strengthening of identity, performed at the federated level, generally requires a component information base to give up some of its autonomy with respect to unilateral updates.

4 A canonical object model

As a framework for the subsequent discussion we briefly introduce a canonical object representation model. We do not intend to present a complete language, but rather to fix ideas of how the different levels of support for identity can be coped with in a canonical object model. For this reason, the object representation model only covers aspects of object identity. For each global object the model basically encompasses a mapping from the object's global identity to the local identity of the corresponding object in a component information base. Furthermore, we only consider global object types that are generalizations of exported types. The inclusion of other object structures (e.g. tuples and sets) does not seem to add any significant new problems to our discussion of object identity. In many respects, our model is similar to the model in [KhCo86].

We assume we are given a set of information bases $\mathcal{D} = \{D_1, \ldots, D_n\}$ that accumulated export a set of syntactically homogenized object types $\mathcal{T} = \{T_1, \ldots, T_m\}$. We also assume that the extension of each exported object type $T_i = \{o_{i_1}, \ldots, o_{i_k}\}$ models a finite set of real world phenomena $\mathcal{P}_i = \{p_{i_1}, \ldots, p_{i_k}\}$ so that o_{i_j} models (some aspect of) p_{i_j}.

We introduce the object model in two steps. The first step assumes that the set of real world phenomena being modelled by the different exported object types, are all disjoint, i.e. $\mathcal{P}_i$ and $\mathcal{P}_j$ are disjoint whenever $i \neq j$. In Section 6 we relax this constraint and extend the object model accordingly.

A global type T is a generalization of a set of exported types $\{T_{i_1}, \ldots, T_{i_k}\}$. The population of objects of type T_j generalized by T, is generally given by a (functional) expression giving a list of objects of type T. Attributes and methods are derived and/or extracted (by suitable functional expressions) from those of the exported types that T generalizes.

A representation of a global object O is a triple $(id, T, (T_j, e_j))$. If the object representation is persistent, id is a globally unique surrogate oid. Otherwise id is a temporary oid unique within the scope of a global transaction only. A global transaction is executed as one or more sessions towards component information bases. T denotes the global type of O, while T_j denotes an exported object type, that T generalizes. And finally, e_j is a (functional) filter that uniquely identifies an object within the population of objects of type T_j. We refer to e_j as an *identifying expression*.

According to our classification of object identity representation, e_j may either be an immutable oid[2], or a (functional) query expression selecting the object based on the value of key attributes. If the corresponding component information base provides sessions oids to its clients, the latter expression might be accompanied by an object variable that outside sessions has the value NULL, while inside sessions is assigned a temporary session oid when the corresponding object is retrieved. However, we may ignore this.

5 Strengthening of identity

If a strong notion of identity is required at the federated level (i.e. the global object representation is persistent), the identifying expression e_j of the global representation of some global object O, must represent a secure object identifier (c.f. above). But if e_j is a query expression based on value based identity, e_j will generally not be secure since the local identity

[2]In our functional database programming language, an oid is syntactically treated as a constant function and hence is also an expression

of objects may change from one session to the other without the global level knowing it. It follows that there is generally a need for making insecure object identifiers secure at the federated level. Whether this is achievable or not, depends on the level of autonomy of the information bases $\mathcal{D}$. For the purpose of our discussion, we say that an information base is *update autonomous*, if it may unilaterally make updates to its objects.

Secure object identification at the federated level can be provided when component information bases are willing to relax their requirement for update autonomy. Either this can be realized by not allowing update of the key attributes of each exported object type, or by somehow keeping the federated information system informed about when such key attributes are updated. In the latter case, this information is used to modify the corresponding identifying expression in the persistent global object representation. Informing the federated level about key attribute updates, can be achieved by keeping an update log for exported types in the component information base, and to which the federated information system (through its data model transforming processors[3] (say)) has read access. This requires, however, a change in the design of the component information system.

Another possibility is to give the federated information system, through its transforming processors, access to the normal transaction log. This does not require a change in the design of the component information system, but might require relaxing the security and privacy requirements. The latter need not be implied if the transforming processor is under the authority of the administrator of the component information base. To monitor the identity of local objects, the transforming processor could filter from the log all local updates on the key attributes of data corresponding to exported data types. This information can be used internally in the transforming processor to maintain a mapping from old to new value based object identifiers thus giving the illusion of immutable (value based) identity. This mapping information could periodically be exchanged between the transforming processors and the federated data manager.

The cost of monitoring and keeping the federated level informed about all updates to key attributes of exported data types, can potentially be overwhelming. This overhead can be significantly reduced or even totally eliminated depending on the degree of persistency of global object identity representation that is required. For example, suppose a disjoint partitioning of the sub-transactions of a global transaction T such that all sub-transactions in a partition are executed at the same site (i.e. information base). Furthermore assume that sub-transactions belonging to different partitions are executed at different sites. If all sub-transactions in a partition can be executed within the same session, all object identities will be secure during the whole transaction. Accordingly there is no need for monitoring this transaction. Note that we assume that changes made to an object's identity by the transaction itself, will cause no problems for secure identification within the transaction.

[3] Transforming processors are the architectural components of a federation providing data model transparency [ShLa90].

As a second example, consider non-isolated transactions with recovery based on compensation and where the primary sub-transaction and its compensating transaction must be mapped to different sessions (c.f. above). For this case, the only requirement is that that the identities of the accessed objects in the primary (sub)transaction are preserved between the two related sessions. This means that the time period for which the fedrated level needs secure identification of an involved object, starts at the beginning of the session to which the primary sub-transaction is mapped, and terminates at the end of the session corresponding to a compensating transaction or at the time it is clear that no compensation will be needed. For a transforming processor, this means that the monitoring of the local identity of an object accessed during the sub-transaction, can terminate at the point in time when the corresponding global transaction commits. The semantics of a commit message for a given sub-transaction will in this case be that the corresponding global transaction has terminated and accordingly no compensation for this sub-transaction will ever be requested from the global level.

Concurrency control at the federated level, based on conflict analysis, requires that object identity must be preserved between global transactions. We are currently studying a concept of global sessions similar to the session concept introduced above, to cope with this requirement. The idea is to group conflicting global transactions within global sessions. The results of this work will be reported elsewhere.

6 Semantic Replication

In contrast to the assumptions made in the previous section, it is often the case that different information bases model aspects of the same real world phenomena. Hence two objects located in different information bases, may model overlapping aspects of the same real world phenomenon. More formally this is expressed by assuming that the two sets of real world phenomena $\mathcal{P}_i$ and $\mathcal{P}_j$, being modelled by two different exported object types T_i and T_j (say), are not necessarily disjoint.

Suppose p is member of the disjunction of $\mathcal{P}_i$ and $\mathcal{P}_j$. If o_i of type T_i and o_j of type T_j both models some overlapping aspects of p, we say that o_i and o_i are *phenomenon equivalent*. We say that a global object O is *semantically replicated* if it generalizes o_i and o_j.

We include *semantical replication* into the object model by extending the representation of a global object O to $(id, T, \{(T_{j_1}, e_{j_1}), \ldots, (T_{j_l}, e_{j_k})\})$ where id is as above. T denotes a global object type whose objects may be semantically replicated. Each pair (T_{j_s}, e_{j_s}) describes one of the semantical replica of O. Each type $T_{j_s}, s = 1, \ldots, k$, is exported by some component information base and belongs to the set of types that T generalizes. The objects identified by the identifying expressions $e_{j_1}, \ldots, e_{j_k}$, are all phenomenon equivalent.

Phenomenon equivalence between objects of different types, is always defined with respect to a generalizing type. Hence for every global object type T generalizing a set of exported types, the conditions for phenomenon equivalence between objects of these exported types, have to be part of T's definition. To be able to decide phenomenon equivalence between a pair of objects o_1 and o_2 of type T_1 and T_2 respectively, it is required that at least some attributes of the phenomenon they model, can be related by an equivalence relation. This is motivated by our belief that comparing attributes are the only means o_1 and o_2 can possibly be automatically related to the same real world phenomenon. Otherwise external assertions about equivalence are needed, to compensate for incomplete information.

We can express this more formally. Suppose T is an object type whose objects might be semantically replicated, and that T generalizes T_i and T_j. Then there must exist a boolean valued function f_{ij}, and attributes a_i belonging to the type T_i and a_j belonging to the type T_j, such that o_i and o_j are considered phenomenon equivalent if and only if $f_{ij}(a_i(e_i), a_j(e_j))$ is true[4], where e_i and e_j are the identifying expressions for o_i and o_j respectively.

We are aware of that updating a semantically replicated object can cause problems of global consistency (if such consistency is required). This is in particular a problem when component information bases are update autonomous. However, due to space limitations, we do not discuss this any further here.

7 Summary

A strong notion of identity is sometimes needed at the federated level, regardless of what canonical data model is used. An object-oriented canonical data model is clearly a good candidate, since object identity is an inherent notion of the object-oriented paradigm. Persistent global object representation is required to provide a strong notion of identity at the federated level. It follows there is a need to provide strong object identity over weak object identity systems. Based on a canonical object representation model, we identified approaches to solutions all having in common that they require component information systems to relax some of their requirements to update autonomy. One solution is based on the ability of the federated information system to monitor any change of identity of objects of exported types with value based identity. An other solution requires that key attributes of object types with value based identity, are immutable. Solutions based on monitoring seems generally preferable since they violate autonomy requirements to a lesser extent than solutions implying reduced update autonomy. The cost of monitoring can be significantly reduced by utilizing knowledge about the actual requirements to global object identity from a given federated transaction model.

Global, generalizing objects may be semantically replicated through phenomenon equivalent objects. For the federated system to be able to automatically identify the semantical replica of a global object, we presented an approach based on the definition of an equivalence relation between pair of objects of any two different exported types.

[4] The notation x(y) denotes functional composition

References

[AbBo91] Abiteboul, S., Bonner, A., Objects and Views, In Proceedings SIGMOD'91, ACM, 1991, pp. 238-247.

[CoLy88] Connors, T., Lyngbaek, P., Providing Uniform Access to Heterogeneous Information Bases, In: Advances in Object-Oriented Database Systems (K.R. Dittrich ed.), LNCS 334, 1988, pp. 162-173.

[ElKa91] Eliassen, F., Karlsen, R., Providing Application Interoperability using Functional Programming Concepts, In: Proceedings EurOpen Spring Conference 1991 (ed. F. Brazier), pp. 249-266.

[ELLR90] Elmagarmid, A.K., Leu, Y., Litwin, W., Rusinkiewicz, M., A Multidatabase Transaction Model for Interbase, Proceeding 16th VLDB, 1990.

[KhCo86] Khoshafian, S. N., Copeland, G. P., Object Identity, Proccedings OOPSLA'86, ACM, Sept. 1986, pp. 406-416.

[KrLK91] Krisnamurthy, R., Litwin, W., Kent, W., Language Features for Interoperability with Schematic Discrepancies, Proccedings SIGMOD'91, ACM, pp. 40-49.

[O2Te91] A Technical Overview of the O_2 System, O_2 Technology, July 1991.

[ShLa90] Sheth, A., Larson, J., Federated Database Systems for Managing Heterogeneous, and Autonomous Databases, ACM Computing Survey, Vol.22, No. 3, Sept 1990, pp. 183-236.