Semantic Equivalence in Engineering Design Databases

Bill Hills
Engineering Design Centre,
Newcastle University
bill.hills@ncl.ac.uk

Barry Florida-James
Engineering Design Centre,
Newcastle University, NE1 7RU
b.o.florida-james@ncl.ac.uk

Nick Rossiter
Computing Science Dept,
Newcastle University
b.n.rossiter@ncl.ac.uk

Abstract

The problems of integrating data from distributed heterogeneous information sources are well known. Much research has concentrated on how to communicate knowledge between several such systems. One of the major issues has been that of identity between related schema. Here we examine the issue of entity identity as it applies to engineering design databases. We describe various techniques for data integration with an example of a prototype system. The suitability of these techniques for engineering design information are discussed. Finally, we show a practical approach to the problem which utilises emerging standards in distributed object technology CORBA and information modelling STEP.

Introduction

With the increase in international collaboration between companies the way design is conducted is changing significantly. Strategic alliances and partnerships often require design activity to be carried out simultaneously between widely distributed design agents. This is particularly the case when the products concerned are large and complex such as aerospace and offshore.

In these large Made-To-Order products there exists many models intended to represent different aspects of the same real world entity. For example, an electical engineer may want to simulate the current characteristics of a pump whereas the structural engineer will want to examine the forces it exerts on neighbouring entities. Hence, the representation of these items in local databases may be so different as to make them unrecognisable but they are inextricably linked by the real world entity.

This leads us to the following question; 'How do we compare two models based on different concepts?'. In engineering the problem is aided to some extent by concepts and modelling being relatively well standardised on established engineering principles. For example a circuit diagram usually consists of standard symbols from a finite set. However, different storage mechanisms may represent the symbols differently so common interfaces need to be defined.

Identity and Naming

Kent[Ken91] states that the most fundamental principle on which modelling rests i*s a one-to-one correspondence between the proxy objects in the database and the entity objects in the real world.* It is this one-to-one correspondence which is assumed in most single database systems, but is shown to breakdown over a multidatabase, that we want to achieve at the federated level.

Traditionally relational database systems have based identity on the presence of a key value. The problems of this are given by Kent as follows:

- Some objects may have no primary relation in which their identifiers serve as primary key.

- The same key might be a primary key in several tables, so that insertion into a table does not necessarily imply creation of a new object. For example *employee benefits* and *payroll* information can be kept in separate tables, each with the same key of *employee numbers*.

Proceedings of the 4th KRDB Workshop
Athens, Greece, 30-August-1997
(F. Baader, M.A. Jeusfeld, W. Nutt, eds.)
http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-8/

It is established therefore that value-based systems do not model identity very well. Object-based systems have a much stronger sense of identity. Eliasen & Karlsen[Eli91] categorise object identity (OID) at three levels:

Value Based Identity - Identity by a key attribute value such as a name

Session Object Identity - Object identity which exists during a transaction

Immutable Object Identity - Object identity which exists across transactions.

Obviously it is achieving the latter form of identity which concerns us. A starting point for this is the selection of a suitable federated or global model.

Global Model

The data model chosen for the integrated design framework must be detailed enough to model attribute information correctly and abstract enough to be of use at the enterprise level. Saltor *et al*[Sal91] suggest a methodology for assessing a model's suitability as a *canonical model* for federated databases. They conclude that not all models are equally as suited to the job and that pre-relational models are of little value whereas functional models and some Object-Oriented models, particularly those supporting views, are better.

Rusinkiewicz *et al*[Rus91] suggest that a better approach than modelling constraints on data models is to model dependencies between them. These dependencies if properly modelled offer a better view of the world as they contain more about the semantics contained within the models. They include information about the state of the data as well as temporal information.

A suitable global object model is described by Eliasan & Karlsen[Eli91] and they point out two major issues from this approach:

- There is a large overhead in mapping identities to the global level.

- It is not always possible to map a local object to a single entity at the federated level.

Eliasan[Eli95] produces an architecture addressing the issue of identity and finds that computed OID maps are not practical unless frequency of local identity update is low. However low update frequency is a feature of engineering design databases.

Integration Approaches

Attempts have been made to generate equivalencies of data in multiple databases. Larson *et al*[Lar89] define attributes by a set of charcteristics. These characteristics are used to define a measure of equivalence. They suggest that a knowledge base could be used to equate attributes based on their mapping functions but this is not applied or tested.Yu *et al*[Yu91] provide a method for automatically relating names in heterogeneous databases. They define an iterative process, dependent on a knowledge base, which associates keywords with each other. The process described is automated as far as possible but requires user interaction to consolidate the associations.

Urban & Jian[Urb91] present a framework whereby heterogeneous schema in data models may be uniformly described. They argue that because of the 'inherent incompleteness of legacy databases' semantic equivalence must in general be obtained by the use of additional *a priori* assertions that are external to the representations under consideration. Urban & Jian show how import and export procedures can be combined with global to local mappings to enhance inter-operability of heterogeneous schema. They suggest that the hardest issue to resolve is that of identity of objects in more than one database.

Pegsasus[Sha95] is a heterogeneous multidatabase management system with which external data sources are registered and import schemas are created to allow data extraction. OIDs are generated for instances of imported types. Currently they are constructed using a prefix associated with the imported type and a suffix from the value set of its generating expression. Imported types are assumed to be disjoint on instances. Therefore the same instance will have exactly the same OID within each type to which it belongs.

Work at Queensland University[Yan95] classifies and organizes correspondences between heterogeneous object-oriented schema. This information resides in a knowledge base attached to each local database. The knowledge base allows remote objects to be treated as local data types and also determines which part of a query is local and which is remote. At Stanford[Wie96] a data warehouse system has been produced based upon CORBA objects and asynchronous messaging. Here a meta datastore is used to resolve relationships across data stores rather than a knowledge base.

The above applied systems all deal with the problem of semantically equivalent objects residing in multiple databases. They approach the problem in various manners and with different degrees of practical success. However, it should be noted that none of them operate in a fully automated manner. It is felt that the best practice in engineering design is to allow the designer to ultimately decide what is equivalent to what. The rest of this paper presents this approach and demonstrates its application in a working prototype.

Presenting the Global Agent

In work at Newcastle, we have produced an implementation of an agent architecture based on the distributed object paradigm. The agent system architecture is of a similar nature to that desribed by Singh & Gisi[Sin96] being based on CORBA components with declarative interfaces specified in IDL. As demonstrated in this work there are limitations with this as IDL is inherently a syntactical language and not suited to modelling semantics.

This is being addressed by the development of a communication language which will facilitate the exchange of knowledge between the agents. This approach is described by Genesereth and Ketchpel [Gen94], where the Agent Communication Language (ACL) is presented. Work is currently underway to produce a subset of this language which is specific to engineering design.

The overall agent architecture is shown in Figure 1. In the figure circles are shown to represent the agents within communicating layers. Communication across the layers is represented by the arrowed lines.

A brief description of the role of each type of agent is given. However, we will concentrate on the role of the global agent in dealing with the issue of semantic equivalence in engineering design databases.

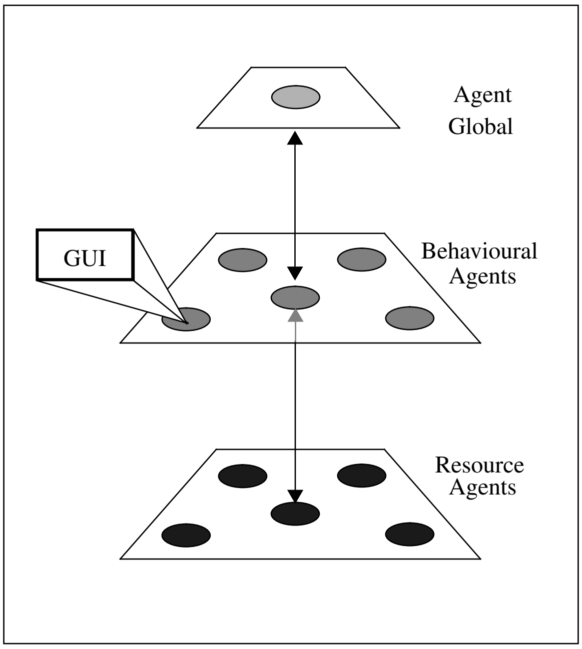

Figure 1: Three-tier Agent Architecture

The agent architecture is a three-tier architecture with the designer introduced through behavioural agents (BA) in the middle layer. The resource agents (RA) act on behalf of the local database. Agent communication is based on clients acting a senders and servers as receivers of messages. At present it is assumed that no messages are lost.

The global agent (GA) has the responsibility for maintaining consistency across the information sources. The assumption is that there is only one active configuration i.e. design moves from one state to another single state until its completion. This may be assumed given that in the overall system, conflict resolution and control management procedures exist to ensure this. It may be required in some instances to have two versions of GAs existing in parallel and this is done simply by the agent replicating itself, however ultimately one will be discarded.

Consider a typical situation where two GAs may be required. An industrial partner, AMEC Process & Energy plc, was uncertain as to whether an offshore platform it was designing would be manufactured from steel or concrete so two alternatives were progressed until this issue was resolved.

The GA has the following commitments or obligations[Sho93]:

- Knowing the location of resources
- Knowing the current configuration of the product
- Controlling creation and deletion of global objects
- Knowing relationships between objects at the global level

Global objects and global relations can only be created and destroyed by the GA. These global IDs are maintained across transactions and access to global objects and relations is only via the GA. As well as knowing the current configuration the GA maintains a history of the product evolution.The following section describes a typical example of the agent behaviour.

Example Behaviour

The GA has got a message from one of the BAs indicating that a change has been made in the local model:

- The BA presents the change by first talking to the local RA (e.g. a comparison algorithm)
- The GA examines its list of current entities and either

  Versions the current configuration and creates a new one

Or

Says I don't recognise this entity and sends a query to the BA: Is this a part of or equivalent to something already existing?

•The BA makes this decision by user intervention (see browse procedure below)

•The BA makes the appropriate request to the GA based on this decision

Browse Procedure

If the GA encounters a name that it cannot resolve it follows a simple given procedure. The GA has a root object and from this root other objects are displayed in a hierarchical manner. The design engineer browses this tree until he finds the component or sub-component with which he is concerned, any new object must either be a part of or equivalent to some other component. This is illustrated in Figure 2. The agent responds to this by either making a new mapping to another local object at the global level or creating a new object with a unique id and a new relationship which places it within the hierarchy. The worst case in this instance is that all objects exist one level below the root object, however, it is felt that design inherently lends itself to this hierarchical model and it is an intuitive procedure for design engineers to use. Thus we have a simple but effective procedure for resolving the issue of semantic equivalence.

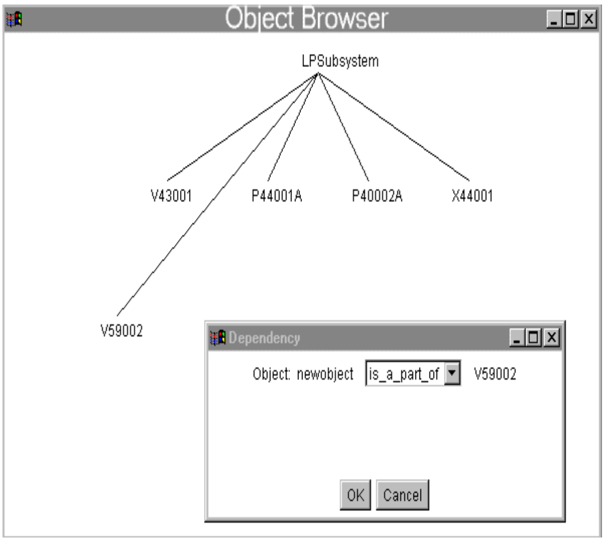

Figure 2: Screen Shot of Browse Procedure

Implementation Issues

The architecture has been implemented as a working prototype to examine the behavioural capabilities of the global agent. The implementation is in C++ and CORBA IIOP (Internet InterOrb Protocol) in the form of Iona's Orbix 2 product. C++ was used in preference to a declarative language such as Prolog. It was thought that C++ gave us the ability to implement the knowledge model through appropriate abstractions and also provided a powerful tool for producing robust maintainable software.

The global agent resides on a Solaris Sparc architecture but simulated behavioural agents have been built on Solaris and Windows NT platforms. The network communication is also not confined to a local area network and we have demonstrated remote operation across platforms in Sunderland and Newcastle Universities.

The CORBA IIOP allows bindings across ORBs and also allow internet access via Java. Java clients have been partially implemented but we found limitations in this. Through Netscape Navigator 3.01 we could not perform file I/O due to security restrictions so we could not cache information at the client side. This is an important feature of our implementation due to the potentially large amounts of data involved in engineering applications. However, the potential for access through a browser is not precluded in our approach and it is intended to utilise this further as the technology matures.

The STEP Standard Data Access Interface (SDAI) [Fow95] is used at the local sites to present access to data in a uniform manner. The application protocol (AP) 231 'Application Protocol: Process Engineering Data: Process Design and Process Specifications of Major Equipment' was used in the prototype as it is concerned with a particular case study from the process industry. Again there are certain limitations with this as some of the local repositories do not map well into the standard AP. We allowed for this by extending the EXPRESS schema to suit our needs, it is obviously hoped that as the data standards develop we can incorporate it more into the system to give a more generic architecture.

Final Remarks

In the field of engineering design it is possible to use STEP and CORBA to provide a standard open architecture. Such an architecture will incur overheads in terms of system development and runtime processing. However, these overheads are minimal when compared to the amount of money and time that can be saved by asserting rules and identifying conflicts earlier in the design process.

The use of an agent architecture allows the incorporation of the designer directly into the system rather than attempting to model his knowledge explicitly. It is felt that whilst automated processes are valuable, invariably the designer likes to exercise control over these. In the prototype described here the designer is asked to resolve ultimately one of the most important issues of semantic heterogeneity, that of object identity. By providing a simple access mechanism it is possible to draw from the designer implicit knowledge and incorporate this into the supporting systems.

The notion of agents passing messages allows us to build in varying degrees of complexity within the system. At one end we have a simple database representing knowledge and at the other a user driven graphical interface representing years of experience from a design engineer. By allowing these to communicate in a uniform and productive manner we hope to derive the best from both.

The global model chosen allows the system to control the creation and deletion of objects. The system also tracks version information and organises the global set through global relations which are used to define views. It is hoped to extend these relations to enable representation of more complex information in order that the occurrence of design conflicts may be identified earlier. The explicit definition of objects and relations at one site means that there is no ambiguity in the system.

It is generally recognised that there are significant increases in performance that can be achieved in the design and manufacturing processes if the supporting systems can be fully integrated. The implications of attempting to achieve integration across distributed heterogeneous networks are complex and raise many research issues. The methodologies outlined in this paper are an attempt to achieve fundamental improvements in industries where the products are Made-To-Order.

Finally, it has been shown that many approaches and methods have been defined for deriving object equivalences in heterogeneous databases. In Made-To-Order products the solution to the problem is facilitated as update frequency is low and transaction times are long. However, there is still the major problem that engineering design systems tend to be highly distributed (across continents) and highly heterogeneous.

References

[Eli91] Eliassen F and Karlsen R “Interoperability and Object Identity”, SIGMOD RECORD, Vol 20 No 4 (1991) pp 25-29

[Eli95] Eliassen F “Managing Identity in Global Object Views”, RIDE-DOM , Taiwan (1995) pp 70-77

[Fow95] Fowler J “STEP for Data Management, Exchange and Sharing”, Technology Appraisals (1995)

[Gen94] Genesereth M and Ketchpel S “Software Agents”, Communications of the ACM, Vol 37 No 7 (1994) pp 48-53

[Kent91] Kent W “The Breakdown of the Information Model in Multi-Database Systems”, SIGMOD RECORD, Vol 20 No 4 (1991) pp 10-15

[Lar89] Larson J, Navathe S and Elmasri R “A Theory of Attribute Equivalence in Databases with Application to Schema Integration”, IEEE Transactions on Software Engineering, Vol 15 No 4 (1989) pp 449-463

[Rus91] Rusinkiewicz M, Sheth A and Karabatis G “Specifying Interdatabase Dependencies in a Multidatabase Environment”, Computer, Vol 24 No 12 (1991) pp 46-53

[Sal91] Saltor F, Castellanos M and Garcia-Solaco M “Suitability of data models as canonical models for federated databases”, SIGMOD RECORD, Vol 20 No 4 (1991) pp 44-48

[Sha95] Shan, Ahmed, Davis, Du and Kent “Pegasus: A Heterogeneous Information management System”, in Modern Database Systems (1995) pp 664-682

[Sin96] Singh N and Gisi M “Coordinating Distributed Objects with Declarative Interfaces”, Lecture Notes in Computer Science 1061 (1996) pp 368-385

[Sho93] Shoham Y “Agent-oriented programing”, Artificial Intelligence, Vol 60 (1993) pp 51-92

[Wie96] Wiener J, Gupta H, Labio W, Zhuge Y, Garcia-Molina H and Widom J “A System Prototype for Warehouse View Maintenance”. Materialized Views: Techniques and Applications, Montreal, Canada (1996) pp 26-33

[Urb91] Urban S D and Wu J “Resolving Semantic Heterogeneity Through the Explicit Representation of Data Model Semantics”, SIGMOD RECORD, Vol 20 No 4 (1991) pp 55-58

[Yan95] Yang J and Papazoglou M “A Configurable Approach for Object Sharing among Multidatabase Systems”, 4th International Conference on Information and Knowledge Management, Baltimore, USA (1995) pp 129-136

[Yu91] Yu C, Jia B, Sun W and Doa S “Determining Relationships among Names in Heterogenoeus Databases”, SIGMOD RECORD, Vol 20 No 4 (1991) pp 79-80