December 1999

TIME: A Case Tool for developing Web-based Cooperative IS

Christophe Nicolle
Université de Bourgogne

Nadine Cullot
Université de Bourgogne

Fabrice Jouanot
Université de Bourgogne

TIME : A Case Tool for developing Web-based Cooperative IS

Christophe Nicolle, Nadine Cullot, Fabrice Jouanot
Equipe Ingénierie Informatique - LE2I - Université de Bourgogne
B.P. 47870, 21078 Dijon Cedex - FRANCE
e-mail : Christophe.Nicolle,Nadine.Cullot,Fabrice.Jouanot@satie.u-bourgogne.fr

Introduction

Distributed web oriented processing environments have created the need for information sharing and transparent data access across heterogeneous information systems. This need stems from the proliferation of network based technologies that provide universal access to a growing number of information sources. Recently, new information system architectures including federated databases, cooperative information systems, and interoperable systems have been advocated in database research for the interoperation of multiple autonomous heterogeneous information systems over distributed networks. According to their architectures, the building of cooperations follows different ways. Nevertheless, many steps are shared. Figure 1.a depicts the required components for cooperative building (network, local schema, models) and shows the dynamic exchanges to allow systems to cooperate independently from their architectures. First, each cooperating system has to define the set of shared information defining a view of its local schema. This schema has to be exported either to a global or common system (Distributed Architecture (Barsalou et al. 1992), Tightly Coupled Federated Architecture (Gardarin et al. 1997) or Mediator/Wrapper Architecture (Garcia et al. 1995)), or to other cooperating systems (Loosely coupled Federated Architecture (Anderson et al. 1993)). In all these cases, this schema is described in terms of local models and it is called Export Schema. In figure 1.a Site A and Site C export their schema to B. Next, if cooperating local systems or global/common systems possess similar concepts to understand all received export schemas from other systems then they can directly manage these export schema. In other cases, it is necessary to use model translation tools. Finally, the distant or global system which receives export schema, translates these schemas in a local homogeneous representation called Import Schema and integrates them in a unify view of the cooperation. In figure 1.a, Site B translates Export Schemas A and C into Import Schemas and integrates them with its own local schema to obtain a local cooperative schema in the data model of B. At the end of this step, each cooperating site has a view of the cooperation through a local cooperative schema. Now, they can query this schema, either using their own local data manipulation language (Federated Architecture (Sheth et al. 1990)), or using a global data manipulation language (Distributed Architecture (Keim 1994)). All these architectures need query and data translation tools to cooperative management. This is a key point for the building and the management of cooperative heterogeneous information systems. Figure 1.b depicts an example of cooperative querying. First of all, the cooperative user query (CQ) is decomposed in sub-queries. Next, these sub-queries are translated using query translators according target data manipulation languages and are sent to target systems (SQa, SQc). Then, cooperative systems send the corresponding answer to the source system (Aa, Ac). At the end, all the answers are connected and formatted by data formatting tools in a cooperative answer (CA). This answer is sent to the user.

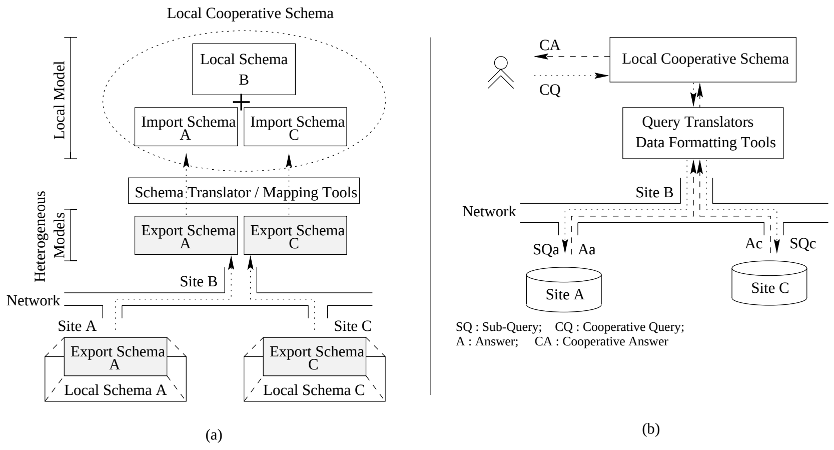

Figure 1. Cooperative System Building (a), Cooperative Querying (b)

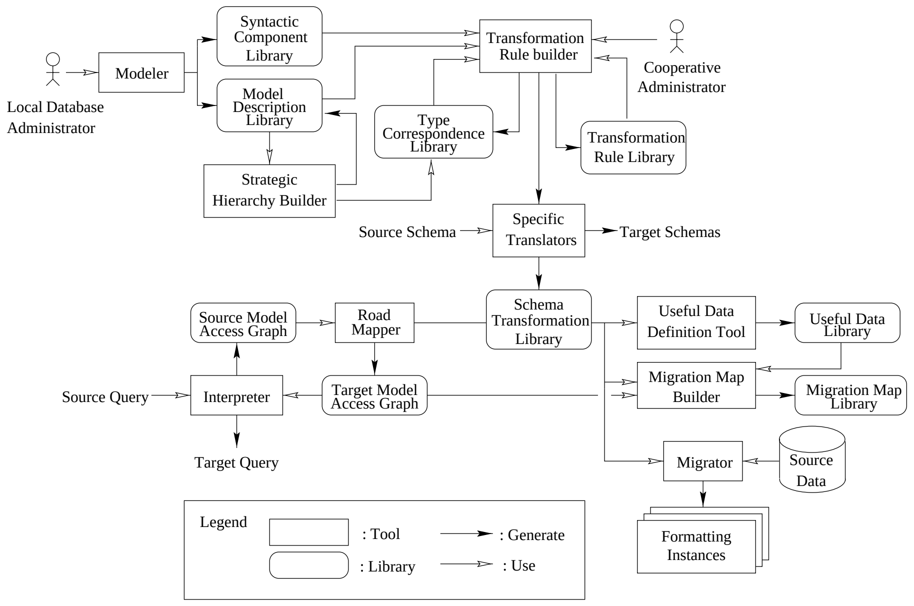
Local Database Administrator
Modeler
Syntactic Component Library
Model Description Library
Strategic Hierarchy Builder
Type Correspondence Library
Transformation Rule builder
Cooperative Administrator
Transformation Rule Library
Source Schema
Specific Translators
Target Schemas
Source Model Access Graph
Road Mapper
Schema Transformation Library
Useful Data Definition Tool
Useful Data Library
Source Query
Interpreter
Target Model Access Graph
Migration Map Builder
Migration Map Library
Target Query
Migrator
Source Data
Formatting Instances
Legend
: Tool
: Library
: Generate
: Use

Figure 2. TIME Architecture Tools

Global Architecture of TIME

TIME provide support for the management of web-based Cooperative IS and takes into account several important characteristics including extensibility and composability (Nicolle et al. 1996). Extensibility requires a translation solution that can easily integrate new data models to the system while composability, which is the ability of an application to dynamically define the subset of data sources it accesses, requires translation among a subset of data models. To achieve these requirements, the semantic resolution and data model translation approaches used in TIME are based on an extensible metamodel consisting of a set of metatypes which act as meta-level semantic descriptors of modelling concepts found in existing data models. These metatypes are organised in a generalisation hierarchy to capture semantic similarities among modelling concepts and correlate constituent data models of interoperable systems. To include new data models in an interoperable information system, the hierarchy of metatypes can be extended with new metatypes which are defined by specialisation of existing metatypes. The generalisation hierarchy of the extensible metamodel provides the foundation for the multi-database design environment. It contains the Schema Translation Module, the Query Translation Module and the Data Translation Module. Figure 2 presents the global architecture of TIME tools.

Schema Translation

Database structures their information in a schema which are built using data models. Usually, in a cooperation, systems do not possess the same data model. To solve this problem, they use tools which allow to translate schemas from source to target models. A reliable technique is the definition of one translator by couple of models. But this technique is not suitable on web based systems where the number of heterogeneous databases can be very large. A less expensive solution consists of using an intermediate model used as a "bridge" between models. Nevertheless, this model allows only the translation between a weak number of predefined models and does not guarantee the extensibility and the necessary opening for web based systems. An alternative to these solutions is to use tools which generate automatically translators between data models. For the translation of schema, we have developed a module specialized in the acquisition of new data models and the semi-automatic building of translators between these models. In order that, we use three main tools. The first tool, called Modeler, allows each local system administrator to define its data model using a description logic language. This tool provides a graphic interface to describe features of data model concepts according to a pre-defined set of knowledge. The Modeler adds to the Syntactic Component Library, the key words of the data definition languages associated with described data models and also completes to the Model Description Library with data model concepts defined in description logic. The definitions of the data model concepts are called metatypes. Then, the second tool called Strategic Hierarchy Builder is used to construct a metatype hierarchy using the subsumption mechanism of description logic. This hierarchy is a specific graph where nodes are metatypes and arcs are specialisation / generalisation links. This work is automatically realised and allows to update metatype definitions contained in the Model Description Library. These definitions can be simplified by analysing inheritance links of the constructed graph. See Nicolle et al 1999 for more details. Figure 2 shows interactions between different tools and libraries described above. The last tool of the schema translation module is the Translation Rule Builder. It uses information contained in the Model Description, Syntactic Component and Type Correspondence Libraries. It allows to define transformation rules between couple of metatypes directly linked in the metatype hierarchy. This tool can be used by the cooperative administrator when all data model concepts composing the cooperation are defined. It proposes a graphical interface to build rules between directly linked metatypes and generates automatically a file in java corresponding to the defined rules. This file is going to complete the Transformation Rule Library. To generate specific translators, the Transformation Rule Builder (TRB) allows to reuse rules and compose them. This composition is realized using a specific analysing graph tool integrated in the TRB. It establishes a path of translation between the metatypes representing the specified models and constructs a java file containing the totality of transformation rules between these models. The resulting executable file is able to read a source schema and translate it automatically into the target models. All the process of transformation is stored in the "Schema Transformation Library".

Query Translation

Our objective is to allow a local user to use its own local language to query distant sites. In order that, it is necessary to define a set of tools able to translate a source query to target data manipulation language. Figure 2 presents the query translation tools defined in TIME. First of all, a source query sent to a distant site has to be translated in an intermediate format, defined according to the schema composition. This task is realised by a tool, called Interpreter, which translates of a source query on an access graph to a target model and inversely. It uses information stored in the Syntactic Component Library and analyses the meta-schema corresponding to the source query to define a corresponding graph. This graph is composed of a set of objects (nodes) and functions (arcs). Next, a Road Mapper transforms this graph step by step according the meta-schema stored in the Schema Transformation Library. This Road Mapper contains data localisation information and can decompose a graph into multiple sub-graphs for multi-querying. Finally, the target query is built from the resulting transformed graph and the syntactic library knowledge corresponding to the target data manipulation language. A couple of interpreter and mapper tools is associated to each heterogeneous data model site of the cooperation. All cooperative queries are sent and received by these tools.

Data Translation

Data translation occurs during the exchange of information between several sites. TIME uses a data translation module to analyse the source query and the associated schema translation. The Data Translation Module uses 1) the Schema Transformation Library generated during the schema translation process 2) the query instances which have to be formatted and 3) the local source database. It is composed of three main tools : the Useful Data Definition tool, the Migration Map Builder and the Migrator tool which are used to create or complete the Useful Data Library and the Migration Map Library. The transformation information stored in the Schema Transformation Library (transformation paths and meta-schema transformations) are used to guide the data translation process which is composed of three steps. First, to reduce working cost and used disk space, a set of useful data is defined. These useful data represent a significant sub-set of data to be migrated. This process is done by the Useful Data Definition tool which builds useful data from the set of key constraints defined in the local schema. The chosen data allow to keep the database coherence during the migration process preserving links between them. Next, useful data are translated according to translation paths to get a migration map. The Migration Map Builder (MMB) uses information from Schema Transformation Library to format useful data according to the target model. This library contains a set of translation paths from source schema to target schema. The MMB generates a migration map which is stored in a specific library (Migration Map Library) used by the Migrator tool. The MMB uses schema transformation rules to construct the transformation process which has to be applied on data. Finally, the migration map is used to extract source data and to directly format them into target data model formalism. Using source schema and corresponding useful data, the Migrator tool extracts data from source database and formats them in the target data model. These data are sent to target systems and they can directly handle them.

Conclusion

In this paper we have described TIME, a case tool to help in the building of web-based cooperative database systems. The TIME environment of translation can be used in many architectures like federated or distributed systems to help in the resolution of semantic and syntactic heterogeneity. It is composed of a set of tools and libraries to translate schema, queries and data. Our future works will focus on extending the metamodel to allow complete modeling of OO model characteristics including the specification of methods and access path constraints.

References available upon request from first author