Gilbert Babin, Cheng Hsu, and Zakaria Maamar. A Distributed Metadatabase Architecture for an Enterprise Scalable, Adaptive Integration Environment. Technical report no. 37-94-424. Rensselaer Polytechnic Institute. Troy, New York, USA. 1994.

A Distributed Metadatabase Architecture for an Enterprise Scalable, Adaptive Integration Environment

Gilbert Babin1
Cheng Hsu2
Zakaria Maamar3

july 1994

1 Assistant Professor, Department of Computer Science, Université Laval, Quebec City, Quebec, Canada, G1K 7P4.
2 Associate Professor, Department of Decision Sciences and Engineering Systems, Rensselaer Polytechnic Institute, Troy, NY, 12180-3590.
3 Master's Student, Department of Computer Science, Université Laval, Quebec City, Quebec, Canada, G1K 7P4.

Abstract

*The research described in this paper extends previous results in systems integration and adaptiveness [1,3] from a single metadatabase environment to a multiple metadatabase environment. More specifically, if each metadatabase contains information describing different operational units within one or more enterprises, the creation of linkages between these metadatabases will effect the integration of these operational units, while preserving autonomy. This need stems from the emerging trends of globalization of enterprises and close coordination among vendors, manufacturers, and even customers, as recognized by such companies as General Electric and General Motors [2]. Results from this problem in their own right also contribute to the unresolved problems of concurrent control and distributed knowledge management in multiple databases and knowledge environments. In the long run, the solution proposed will enable multiple enterprises to conclude commercial alliances that will be enforced automatically through their integrated information systems. Many approaches can be suggested, Intra-Enterprise, Inter-*Enterprise and Hybrid; their use depends on the kind of integration needs to accomplish.

I. Introduction

Consider a large manufacturing enterprise with multiple divisions, each of which might have its own set of application systems. Furthermore, these distinct application systems might utilize heterogeneous information models due generally to: (1) various user needs, (2) various hardware and software platforms, and (3) various physical and logical structures.

For this enterprise to be successful, it must be able to manage and integrate all its information resources. The research we propose provides a framework to enable growing enterprises to manage and possibly integrate thier information resources, as they evolve with another partner. Succinctly, the unique requirements of multiple systems in enterprise information management may be summarized as follows:

- Scalability. The total enterprise environment must be expandable and allow incremental development, such that the integration can start with a small part of the enterprise and gradually extend to the rest of the organization (even to other organizations) over time, without loosing operational continuity and structural integrity;
- Adaptability. Systems that use either standard or non-standard technologies as well as new and legacy systems, can be incorporated into the integrated environment in a seemless way without causing any disruption to the existing architecture;
- Parallelism. The multiple systems must be able to operate concurrently while achieving synergism for the enterprise, without requiring global serialization or similar synchronization mechanisms imposed on any instance-level transactions;
- Autonomy. Local systems in the integration need to have the flexibility to be designed, constructed, and administered independently by local management alone, without having to conform, nor later convert, to a global schema.

The metadatabase work at Rensselaer has focused on creating an enterprise environment fulfilling these requirements. The proposed solution entails the following basic elements:

(1) An enterprise information model: this model globally represents all local data models and their contextual knowledge in the enterprise with a metadata-independent structure which, when put online, allows all local models and other metadata contained in it to be added, deleted, or modified through ordinary metadata transactions (just like database transactions);

(2) An online (independent, but sharable) metadatabase: this metadatabase implements the enterprise information model, and comprises a scalable hierarchy of mini-metadatabases for any scope of local functions in a (expandable) client-server manner;

(3) A concurrent architecture for execution: this architecture supports concurrent processing of local systems with localized distributed control knowledge.

Together, they amount to a metadatabase-supported, rule-oriented concurrent systems solution to the enterprise information management problem. Previous results have focussed mainly on adaptability, parallelism, and autonomy [1]. This paper presents new approaches which consider the scalability and the Distributed Artificial Intelligence (DAI) issues.

At the present time, the persistent applications are becoming more and more "open" in terms of topology, platform and evolution, thus creating a need for integrating them [5]. The integration approaches traditionnally proposed assure the interoprability at the application level via a global schema or manipulation language[6]. Furthermore, most of these (approaches) have provided support at the representation and specification levels with little or no concern to scalability. The scalability concept is generally of concern in different fields of research, starting from the distributed system to the interoperability of databases via the services provided on wide-area networks. Scalable systems are systems that work well on small and large scales.

The development of powerful concurrent computers and the proliferation of multinode computer networks help the researchers to assume that the majority of problems can be resolved by distributing them on a collection of agents, as it occurs with DAI. Multiagent systems are concerned with coordinating behavior among autonomous intelligent agents, how they coordinate their knowledge, goals, skills and plans jointly to take decisions or to solve problems [7]. Our primary objective is to integrate the metadatabase environment and build a cooperative system. A Cooperative Knowledge Based System CKBS is considered in [4] as a collection of autonomous (heterogenous) KBS, refered to as agents, which are capable of interacting with each other. A knowledge base is a collection of data and rules, and therefore a database system can also be an agent. The cooperation is the ability of one agent to work with another to solve a problem; it implies the coherence of the system and, as studied in [4], is concerned with: (1) knowledge consistency across agents, (2) reliability of the overall system, (3) integration of subsolutions, and (4) global performance improvement.

The paper is organized as follows. In this section, we first presented an overview of the research problem at hand. Section II presents the current architecture put forward in the Metadatabase approach. Using the concurrent architecture as a basic framework, we design two basic cooperation schema to allow for scalability, in Section III. In Section IV, we propose the Rule-Oriented Programming Environment [1,3] (ROPE) as a solution to develop the newly defined architecture components. Finally, we conclude the paper in Section V.

II. The Metadatabase Concurrent Architecture

The concurrent architecture is depicted in Figure 1. The metadatabase itself (a rigorously constructed collection of enterprise metadata) provides an integrated enterprise model for the multiple information systems, their databases, and the interactions between the different systems;

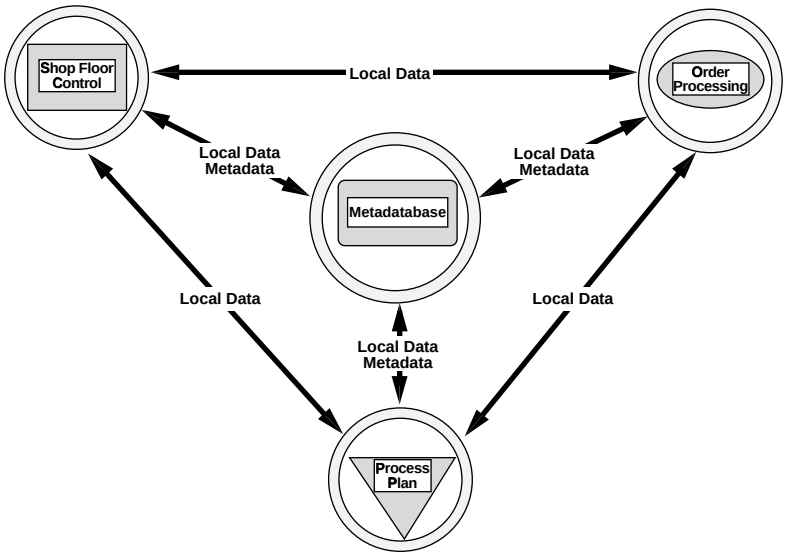

Figure 1 : The Concurrent Architecture Using a Metadatabase

i.e. the information contents and their contextual knowledge. The metadatabase approach (1) uses the enterprise model to assist end-users performing global queries free of both technical details and a hierarchy of integrated schemata, (2) distributes the contextual knowledge to empower these local systems to update data and communicate with each other without central database control, and (3) incorporates legacy, new or changed local models into its generic structure of metadata to support evolution without system redesign or recompilation. The shells in the concurrent architecture, therefore, implements the distributed (localized) knowledge which, in turn, is managed by the metadatabase. The metadatabase schema is a generic model named GIRD (Global Information Resource Dictionnary) [1], and is used to represent the global data model and to integrate the knowledge for the local applications.

The metadatabase approach does not prescribe any actual implementation of the concurrent architecture. However, any instantiation of the concurrent architecture must provide a scalable, adaptable, parallel, and autonomous environment for the integration of the enterprise's application systems. One such implementation of the concurrent architecture was developed at Rensselaer, namely ROPE [1,3]. ROPE defines the structure of the concurrent architecture's shells, their behavior, and how they should be managed by the metadatabase. The main concern

in the design of ROPE was to minimize coupling with the local environment, hence maximizing portability and assuring that the local systems stay autonomous.

III. Basic Architectural Constructs

As the enterprises become larger and as alliances are made with other enterprises, this model of the concurrent architecture might empede on the scalability of the proposed approach. Hence, more general architectures must be defined to enable an adaptable and scalable environment for enterprise information management.

The majority of integration approaches uses a central mecanism to control the global transaction system. This kind of architecture presents the following drawbacks: (1) possible dead-lock of local applications, if the central controllor breaks down, and (2) bottleneck created by a central controllor if the adressed number of transactions is important. The approach that we consider uses the metadatabase concept and the development method TSER as an integration means for preserving the same data models and the same distributed information rules. More specifically, we want to consider a global environment with a set of distinct local environment; each local environment can be represented by its own metadatabase and has to be integrated with the remainning ones. This new approach suggests (1) the development of a new global metadatabase, resulting from local application and metadatabase integration (Intra-Enterprise Approach) or (2) the design of a communication process between the metadatabases for cooperation (Inter-Enterprise Approach). The high level metadatabase and the cooperative architecture are used to enable a global integration strategy, while the low level metadatabases serve to strategically group the local functionnal systems. Based on these two approaches, we define two basic architectures: (1) a hierarchical architecture of metadatabases (Intra-Enterprise Approach) and (2) a cooperative architecture of metadatabases (Inter-Enterprise Approach).

A. Hierachical Architecture of Metadatabases

In large-scale enterprises, the concurrent architecture is expanded to comprise a hierarchy of multidatabases, as illustrated in Figure 2. A hierarchical architecture of metadatabases presents

similarities with the approach developped in [1]; there is a direct correspondance between them: on the one hand the local MDBs and applications, and, on the other hand the local applications integrated in [1]. The same correspondance can be make between the global MDB and the MDB which is the result of the local integration in [1].

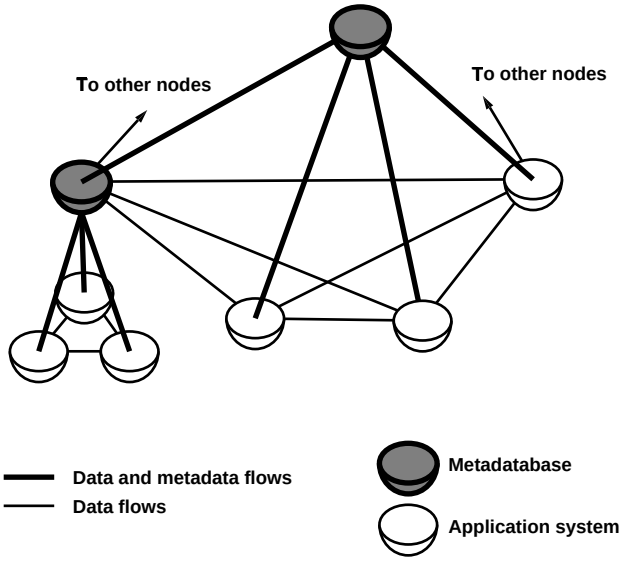

Figure 2: A Hierachical Architecture of Metadatabases

Using this architecture, several application systems are represented in a mini-metadatabase, which is in turn represented in the main metadatabase. The mini-metadatabase becomes the gateway to the application systems it represents. In this architecture, the amount of metadata stored in the main metadatabase is not reduced; it will still contain a model of all the application systems located below it (recursively). This architecture is useful to partition the application systems into sets where the connectivity among application systems is large, hence concentrating their information flows within the same network segment. Also, it presents the following features:

- integration between MDBs and applications;
- varied data exchange:
  - local data between applications;

  - local data and metadata between applications and their local MDBs;
  - global data and metadata between applications, local MDBs and the global MDB.
- a global MDB resulting from the integration which will:
  - provide an integrated model for the multiple application and MDB models;
  - enable the functionalities of the global system;
  - integrate a set of MDBs and applications without causing any disruption to the system.

Two sibling metadatabases (i.e., two metadatabases under the supervision of the same metadatabase) do not share any information. Their metadata content is limited to the description of the application systems and/or metadatabases located below them in the hierarchy. The global data and metadata management will be achieved by the global MDB.
The hierarchical architecture of metadatabases presents the following advantages:

- similar method to [1];
- system functions are well defined:
  - local integration made by the local MDBs;
  - global integration made by the global MDB.
- schema specifications of the local MDBs require a minimal number of changes;
- possibility to formulate a request from three different levels: applications, local MDBs and global MDB.

and also some drawbacks:

- global functionning of the system depends on the abilities of the global MDB;
- different kinds of data and metadata exchange;

- heterogenous integration:

  Integration $_{local}$ = {App$_1$, … , App$_n$}

  Integration $_{global}$ = {App$_1$, … , App$_p$, MDB$_1$, … , MDB$_m$}

B. Cooperative Architecture of Metadatabases

When security is of importance and/or only part of the information stored in a certain system should be accessible to the other system (e.g. as is the case with alliances among enterprises), the cooperative architecture of metadatabases (see Figure 3) will be of use.

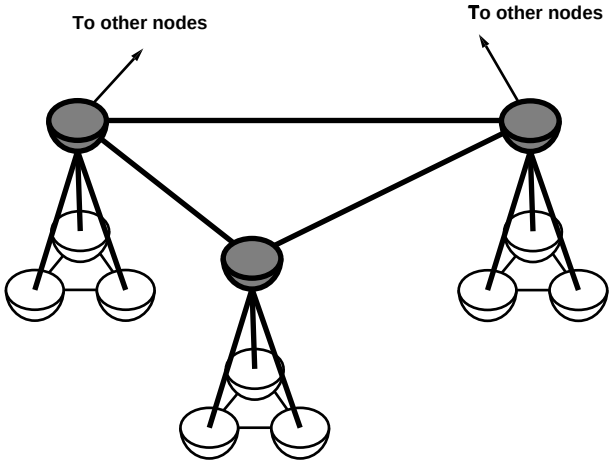

Figure 3 : A Cooperative Architecture of Metadatabases

This kind of integration can be considered as a multiagent system, where the local MDBs are assimilated to autonomous agents. The objective is to establish a cooperation between them. This is achieved by a communication device, allowing a metadata exchange flow. The agent actions and interactions are at the root of this kind of system (cooperative one). Again, each metadatabase will contain a model of all the application systems located below it. However, it might only have partial information from the neighboring metadatabases. This allows for selective sharing of information. A given metadatabase will transit metadata to the other metadatabases on a-need-to know basis. This way, the information model of a given metadatabase is composed of the integration of its local application systems and the shared models from the other metadatabases at the same level. The integration of these metadatabases

could be done by using the Intra-Enterprise approach, i.e, using a global MDB. However, this approach presents a number of difficulties in this context: (1) the data confidentiality for each enterprise would not be guaranteed, (2) the localization of the global MDB is problematic (problem of responsability and physical location), and (3) risk of autonomy loss for the local MDBs; hence, the interest of using cooperating MDBs. The cooperation in a multiagent system must assure the coherence between the structure components (the agents) by insisting on (1) contextual knowledge coherence between agents, (2) reliability for the overall system, and (3) capacity of integrating partial results. All the MDBs must develop global and partial views [4]. The partial views which correspond to the local integration are obtained by using ROPE [1], while the global views which correspond to the global integration are the result of the cooperation. Therefore, for each view we will have different kinds of knowledge: (1) local knowledge matching the local views, and (2) global knowledge matching the global views.

The cooperative architecture of metadatabases presents the following advantages:

- approach used for multiple enterprise integration;
- new design approach is proposed by considering the MDBs as an intelligent agents;
- possibility of using the external information resources of the enterprise;
- reduction of integration complexity by using DAI.

and the following drawbacks:

- difficulty to realize the integration if the number of MDBs to integrate is important:

  L=n*(n-1)/2;

  L: number of links;

  n: number of agents;

  The link number increases proportionnally to the square of the number of agents;
- metadatabases having double functionnalities: local and global integration;
- risk of conflicts between agents, thereby necessitating the development of a negociation protocol;

C. Mixed Architecture of Metadatabases

The two architectures can be used concurrently, thereby creating a mixed architecture of metadatabases. The same rules of metadata distribution apply, depending on the relationships between the different metadatabases. The goal of the Mixed Architecture approach is to realize an internal integration between the enterprise components (directions, departments, ...) as well as to reach an integration between enterprises. It presents pratically the same advantages and drawbacks as the approaches seen before. A large number of development scenarios can be associated to the hybrid approach, depending on the enterprise structures and the importance of data flows between applications. For instance, we can consider:

- two enterprises developing global MDBs by using the Hierarchical approach. They will be integrated by the cooperative one;
- two enterprises developing local MDBs by using the approach developed in [1]. They will be integrated by the cooperative one.
- etc

These scenarios show the variety of integration cases and insist on the requirements of system information management seen before.

IV. ROPE: An Implementation Approach

The Rule-Oriented Programming Environment (ROPE) method develops the shell technology needed for the concurrent architecture of the metadatabase integrated enterprises [1]. It defines (1) how the rules are stored in the local shells, (2) how the rules are processed, distributed and managed, (3) how the shells interact with their corresponding applications, (4) how the different shells interact with each other and (5) the architecture of the shells. ROPE assures system integration by processing the global and update requests at the local applications. Five types of triggers are defined to allow the shells to execute these operations: (1) time-triggered, (2) data-

triggered, (3) program-triggered, (4) rule-triggered and (5) user-triggered. To handle the MDB functionnalities [8], the ROPE dynamic structure presents three classes of languages:

- shell-definition language allowing shell creation;
- modeling and rule language helping the description of the MDB and the shell knowledge;
- message protocol and language using for the communication accross the different applications.

These languages hold up the transactions between the (ROPE, local application system) pair for rule execution and knowledge management operations, and the (ROPE, metadatabase management system) pair for maintaining the MDB schema. Presently, in the context of ROPE and the concurrent architecture, the use of the Metadatabase Management System (MDBMS) is limited to two tasks: (1) global query processing for application users by using the MDBMS or the local shells and (2) knowledge management for creating a new shell. The transactions defined in ROPE are in relation with *the processing of rules by the shells, the local data queries, the shell structure modification* and *the global data queries*. In addition to these transactions, the hierarchical and cooperative architectures demand new mechanisms to assure local and global autonomy for the concurrent architecture for applications. These mechanisms will enable (1) data exchange within and outside the enterprise, (2) the establishment of communication links between metadatabases, and (3) the definition of new functionnalities for the shell.

V. Conclusion

The different integration approache s presented above are only a means to make the integration of external and internal information resources possible. These resources are used to establish a contact between economical partenars. The MDB concept reduces the complexity of the

integration problem by: (1) suggesting a simplified design for the system architecture; and (2) integrating the concept of knowledge based system with distributed data management.

We presented two approaches for assuring this type of integration. One approach suggests *a Hierarchical Architecture* and another suggests *a Cooperative Architecture* of metadatabases. The resulting global architecture is a combination between metadatabases, applications and communication links. Within an enterprise, the hierarchical architecture allows for the gradual integration of the different information systems. Furthermore, it allows for highly cooperative application to be grouped under the same metadatabase. Across enterprises, the cooperative architecture enables enterprises to share information resources to effect collaboration.

References

[1] Babin, G., Adaptiveness in Information System Integration, Doctoral thesis, Department of Decision Sciences and Engineering Systems, Rensselaer Polytechnic Institute, Troy, New York, USA, August 1993.

[2] Hsu, C., G. Babin, M. Bouziane, W. Cheung, L. Rattner and L. Yee, “Metadatabase Modeling for Enterprise Information Integration,” Journal of Systems Integration, January 1992, 2(1), pp. 5-39.

[3] Hsu, C. and G. Babin, “A Rule-Oriented Concurrent Architecture to Effect Adaptiveness for Integrated Manufacturing Enterprises,” International Conference on Industrial Engineering and Production Management (IEPM ’93), Mons, Belgium, June 1993, pp. 868-877.

[4] S. M. Deen, "A General FrameWork for Coherence in CKBS," Intelligent Information Systems, 1(3), september 1, 1992.

[5] Franchitti, J.-C and King, R, " Amalgame: a Tool for Creating Interoperating Persistent, Heterogenous Components," Advanced Database Systems, 1993, pp. 313-336.

[6] Christine Collet et Michael N. Huhns et Wei-Min Shen, "Resource Integration Using a Large Knowledge Base in CARNOT," Computer, December 1992, pp. 55-62.

[7] Alan H.Bond and Les Gasser, "Readings in Distributed Artificial Intelligence," Morgan Kaufman Publishers, INC. San Mateo, California 1988.

[8] Cheng Hsu and Laurie Rattner, "Metadatabase Solutions for Enterprise Information Integration Problems," Database, Winter 1993.