A Concept of Datamigration in a Distributed, Object-Oriented Knowledge Base

Oliver Schmid
Research Institute for Robotic and Real-Time Systems,
Department of Computer Science,
Technical University of Munich, Munich
e-mail: schmido@informatik.tu-muenchen.de

Abstract

This article presents the development of a concept for the distribution and migration of data to several databases within a distributed knowledge base. Thus a concept for a load-dependent and application-specific migration of data is introduced which takes into consideration the number and sort of accesses to the data. The main part of the migration method is a distribution function, which generates an optimal distribution of data on the databases according to the defined aims. It uses all information of the profile of accesses and evaluates this information with the help of several decision-parameters of different priority. A quick reaction on changes in the profile of accesses is guaranteed by a cycle-protocol, which consists of three parallelly running phases with a sort of pipeline structure.

1 Introduction

Market demand for more complex products with high quality and individual design requires complex manufacturing systems. This creates problems in managing production processes as well as in planning and coordinating the production flow. The use of intelligent systems like autonomous robots can reduce the complexity of the production process, because intelligent systems are able to handle complex tasks independently or in cooperation with others. This increases the degree of freedom for a flexible organization of the manufacturing process. On the other hand this method leads to a large amount of data to be handled. Cooperation and communication tasks as well as the locomotion of the autonomous systems cause a lot of data which must be stored and evaluated. The object-oriented representation formalism with its concepts of abstraction of data, class hierarchy and inheritance mechanisms provide an efficient method for modeling this information in a system for data storage. But also quick reaction and fast cooperation of autonomous mobile systems, like transport vehicles and mobile robots, are required. Speed limits of hardware components especially during disk-operations, hence necessitate the development of a specialized system to bypass these limitations.

Regarding these aspects a distributed, parallel knowledge base has been developed in the subproject Q6 of the Sonderforschungsbereich 331 (SFB331) '*Information Processing Concepts for Autonomous, Mobile Robots*', which is a joint research project sponsored by the *Deutsche Forschungsgemeinschaft*. The knowledge base has several advantages like the object-oriented data-storage mechanism, the possibility of dynamically changing the object-structure during runtime, modeling of procedural knowledge through demons attached to specific objects, active components for automatically sending changed knowledge to interested application programs, multiuser access, secure and permanent data-storage and a distribution concept which allows cooperation between different local knowledge bases ([9]). The latest version of this knowledge base additionally improves the performance of service calls from the application programs by using different concepts for a highly parallel data access ([3], [7]). Thus the data is distributed to several parallel working databases. However, parallel data access and an optimal load-balance of computer and databases highly depends on the distribution of the data to the databases. To guarantee an optimal distribution of data during the whole runtime of the knowledge base it is not sufficient to distribute the data only once at the beginning. On the one hand the data is distributed without exactly knowing the access structure of the application programs. On the other hand changes in the access pattern of the service calls can cause completely different demands on the distribution. Thus it is necessary to develop a migration concept, which creates an optimal distribution of data and initiates a corresponding migration by a load-dependent and task-specific mechanism. The concepts worked out for the migration of data are explained in this paper.

Chapter 2 introduces the concept of the distributed, object-oriented knowledge base of the SFB331. Chapter 3 then discusses the migration concept in detail.

The aims and the parameters of our distribution function are presented which should generate an optimal distribution of data by using the information of an access-protocol of the databases. But also several strategies to initiate the migration mechanism and a three phase protocol in which the migration mechanism could run are introduced. Finally, chapter 4 summarizes the paper and gives an outlook on further research on this topic.

2 The SFB331 knowledge base

This part introduces the basic concepts of the knowledge base developed within the SFB331. Further information on the different topics of this concept can be gained by reading selected items out of [3], [5], [7], [8], and [9].

2.1 The knowledge model

The knowledge base was realized by using an object-oriented data-model in order to give the knowledge within a manufacturing environment a uniform structure. The knowledge engineer can model the declarative knowledge of the autonomous systems, i.e. data, facts etc., in an object structure. The object structure consists of a class hierarchy with single and multiple inheritance between the classes. The real objects are represented as instances of the classes. The properties of the objects are stored in attributes and facets. The knowledge engineer can model the procedural knowledge of the autonomous systems, i.e. algorithms, heuristics etc., by means of procedures (so called demons) attached to the objects of the object structure. The knowledge engineer has synchronous and asynchronous demons at his disposal. Synchronous demons, like *if-added-before* and *if-added-after* are executed in a knowledge base transaction sequentially using in the process the results given by the demons. Asynchronous demons, like *if-added-async* run in parallel to the knowledge base transaction without taking the result into account. A detailed description of the knowledge representation formalisms of the local knowledge bases is given in [5] and [8].

2.2 Architecture

Real-time abilities are a basic requirement for the knowledge base. The architecture of the knowledge base reflects this requirement by supporting a highly parallel execution of the provided service calls.

Within the knowledge base a two-process architecture is realized. This leads to a client-server structure between the application programs and the kernel process of the knowledge base. The clients, i.e. the application programs, send their service calls to the server. The server works the given request and passes the result back to the application program. Several parallelly working kernel processes exist in order to obtain a highly parallel execution. All kernel processes are identical and have the same functionality. This allows a completely free assignment between application programs and knowledge base kernel processes. Furthermore, every kernel process of the knowledge base is connected with several kernel processes of the underlying database system. These kernel processes manage the access to the knowledge stored in the databases. The number of databases and their distribution to the existing computers is freely determinable while initializing the knowledge base.

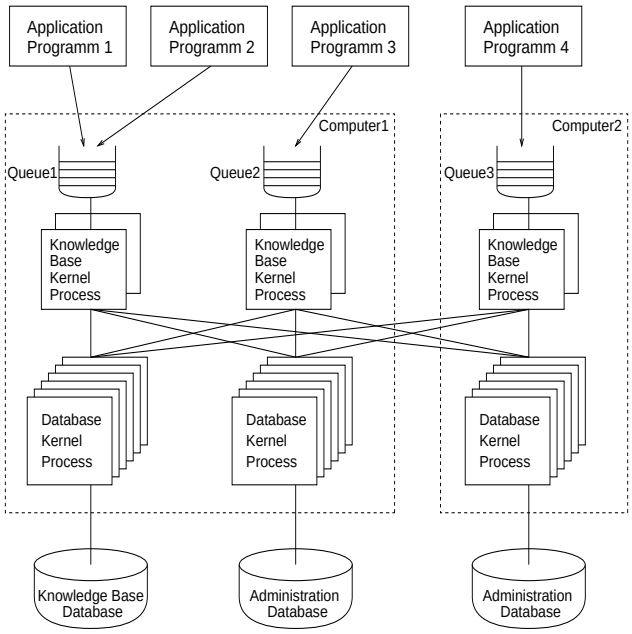

Figure 1: Process structure of the knowledge base ([3])

Figure 1 shows a possible structure of processes of the knowledge base. There exist four application programs which are served by six kernel processes of the knowledge base distributed to two computers. In addition there are a corresponding number of database kernel processes which serve the queries of the knowledge base kernel processes to the databases.

2.3 Data storage and access

The focus of increasing the real-time capabilities of our knowledge base was laid on enhancing the throughput of knowledge base requests as well as on fast write requests, guaranteeing a safe data storage. For this reason a highly parallel data storage system has been developed. This system increases the throughput by allowing our knowledge base kernel to work on incoming function calls in parallel. We use two different kinds of

parallelism. Using the *inter-service parallelism* on the one hand, we are able to work on different function calls in parallel. *Intra-service parallelism*, on the other hand, enables us to work in parallel on a single function call. That means, if several queries are necessary to solve a certain function call in the knowledge base kernel, we try to pass each query to a different database server. Afterwards the results are combined in the knowledge base kernel. To enable several database server to work on the data in parallel, we divided the data into blocks, the so called fragments, which may be accessed in parallel adhering locking protocols. For example, our fragmentation distributes the values of an attribute only for a certain number of the instances to a single database relation. That means the highly parallel data storage system divides the data both horizontally and vertically. Additionally, the relations are distributed to several databases on different computers. In the version which we have already realized, the assignment of relations to different databases is done by hash tables and random distribution ([3]). We use this hash table and an administration database in order to find the relation and database where a certain tuple is stored. The administration database does not become a bottleneck for the parallel execution of the service calls, because it exists redundantly on several computers and there is only non-blocking reading access by those service calls that occur mainly while running a factory.

3 Migration concept

Migration control in distributed systems is mainly known from the area of process-migration and distributed job-administration ([1], [2], [4] and [10]). Some techniques also consider the migration of data between different knots and give algorithms for the data-migration and process-migration (e. g. [4]). In particular [10] presents a model for load-distribution which regards directly the matter of data-distribution in a distributed database system.

Classification schemes are often the basis for the development of specific procedures and mechanisms of migration. The migration concept presented in this paper also employs a classification scheme for migration in distributed, object-oriented systems which was developed in [6]. This scheme was adapted to the matters of the knowledge base in order to create an optimal migration mechanism.

Figure 2 shows the aspects of classifying a migration mechanism within the knowledge base. First of all, the different aims which must be achieved by the migration have to be defined. Then, it is necessary to determine the information necessary to deduce an optimal distribution of data (system-related/task-related, a-priori information/run-time information). The next step is the examination whether the initiation of the migration should be time-related or event-related, or even a combination of both. In order to find an optimal distribution, a strategy must be developed which evaluates the collected information and which regards the result of the evaluation while deciding about a new distribution of data. The decision phase could use heuristic, optimal and suboptimal mechanisms with regard to the respective aims. The structure of the cycle protocol is tightly related to the realization of the migration mechanism. Thus it plays an important role concerning the efficiency of the migration mechanism. In general two phases within the cycle protocol can be distinguished, the phase of gaining information and the phase of decision-making.

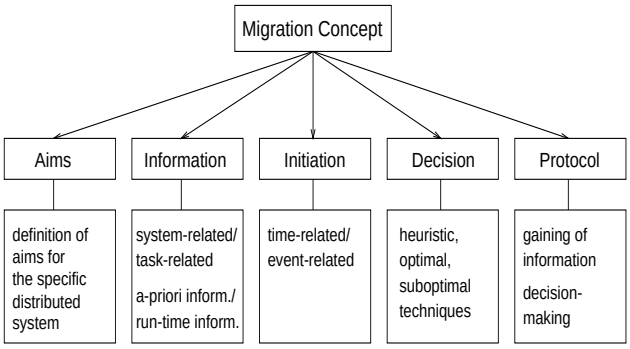

Figure 2: Classification scheme for migration

3.1 Aims

The following aims should be achieved by data-migration between the databases of the knowledge base:

- *Increasing the parallelism of the service calls.* Several service calls can be treated in parallel if the inter-service parallelism and intra-service parallelism are supported by using a good distribution of the data. This reduces the response delay of the service calls and leads therefore to an increase of the throughput of our knowledge base
- *Even distribution of the load.* Distributing the load evenly affects both the load of the databases (amount of stored data and rate of accesses) and the load of the computer as a whole (databases per computer).
- *Reduced costs for communication.* A good distribution of the data stored in the computers at exactly the physical places where they are used most probably reduces the rate of communication between the different computers.

The main priority in our realization was laid on the first aim. This causes a very high rate of really parallel accesses to the different databases which leads to an increasing performance of the knowledge base. The

second aim avoids that a database or a computer becomes a bottleneck within the distributed system of the knowledge base. In practical operation the second aim of an even distribution of the load on the computers has only a slight effect on the performance of our knowledge base, because normally the different databases of a knowledge base are distributed to the number of available computers. An even distribution of the databases already produces a nearly optimal distribution of the load on the computers, as far as it can be influenced by our migration method. A similarly small effect has the third aim. If every database of our knowledge base has its own computer the whole communication is done using the network. But this kind of distribution of the databases makes our knowledge base still more efficient than the effort to reduce the rate of communication and using less computers.

Beside these main aims there are some more aims to be regarded within the migration mechanism. These aims are integrity of the data during migration, high availability of the data, parallel and transparent work of the migration mechanism to the knowledge base and the definition of a narrow interface between the migration mechanism and the knowledge base.

3.2 Information

System-specific information has no relevance for the migration mechanism. The load of the computers and the databases could be evaluated by application-specific information as well and communication costs are negligible (see 3.1). Thus only application-specific information should be taken into consideration as input parameters of the migration method. This information is stored in a parallelly recorded profile of accesses of all application programs. It provides the following information for every access to a database:

- A *time stamp*, showing the date on which the query was given to a certain database.
- The *database* accessed.
- The *computer* holding the data base.
- The *relation* touched by the query.
- The *tupel* influenced by the query.
- The *locking status* of the database required for this kind of operation.

With this information gathered during runtime and a-priori knowledge of the knowledge engineer about the access structure our migration method is able to react in a flexible and quick manner to reach an optimal distribution of the data with regard to the aims.

3.3 Initiation

The initiation of the migration could be time-related and event-related. Time-related in this case means an initiation at predefined times or within a given time-interval. More flexible for the aims of the knowledge base is an event-related initiation. There exist two variants of this sort of initiation, the load-dependent and the task-specific initiation. The load-dependent initiation determines the current load (e.g. by evaluating the profile of accesses). If the load exceeds certain limits the migration mechanism is started. The task-specific initiation takes into account that every application has a specific access pattern to the data corresponding to its job. If this pattern is known a-priori, the migration can be started before or as soon as a certain application program is executed.

3.4 Decision

Parameters for decision. The most important part of the evaluation of information and decision-making is a distribution function. This function gets the recorded profile of accesses as input and computes an optimal distribution of data. In a basic approach to get this distribution function the different parameters for evaluation of the input-data are determined (decision-parameters). While computing a new distribution of data every parameter is regarded with a certain weight corresponding to its importance. The following table gives an overview over these parameters and their weights. A weight of 5 is the highest weight which corresponds to the highest importance, whereas 1 is the lowest weight and therefore represents the least important parameter.

Parameter	Weight
Inter-service parallelism	5
Load of the databases	5
Restrictions of the databases	5
Intra-service parallelism	4
Cost for migration	3
Load of the computers	2
Cost for communication	1

The inter-service parallelism is one of the three most important decision-parameters. In order to maximize this sort of parallelism it is necessary to distribute the single relations in a way that allows a real parallel or at least quasiparallel access to the data by the different application programs. The importance of this parameter results directly from the defined aims of the migration concept.

An even distribution of the database-load is tightly connected with the service-parallelism. Moreover, it avoids an overload of the different databases which otherwise could result in collisions during access or even in errors of the database. Therefore this parameter is as important as the inter-service parallelism.

The restrictions of the underlying database system, like locking status during database-access or the

maximum number of simultaneously active queries per kernel-process, belong to the most important parameters as well. Not following these restrictions will cause deadlocks during the access to the databases or errors of the databases. Furthermore, the information of the locking status simplifies searching a data distribution which supports the service-parallelism.

The intra-service parallelism requires a distribution of data that allows parallel accesses to the databases *within* a service call. This sort of parallelism appears mainly in the definition services of the knowledge base. However, in real applications of the knowledge base definition services occur only rarely, because the knowledge engineer creates the different objects before the application runs. Thus the intra-service parallelism has a lower weight than the inter-service parallelism.

The whole migration mechanism must run parallelly to the usual accesses to the knowledge base. Nevertheless, locking mechanisms might block writing access from other application programs to the databases. A possible lack in availability and a heightened load of the net reduces the performance of the knowledge base, too. These losses can be minimized by starting the migration during times of low activity, but those times could not be guaranteed. Thus the cost for migration should be taken into account with a weight of 3.

Within our migration concept the load of a computer composes out of the rate of accesses and the rate of data stored in a database determined for all databases on this computer. An even distribution of the computers load has only a slight effect on the performance of our knowledge base. This is based on the fact, that a clever distribution of the databases to the available computers already results in a nearly optimal distribution of the computers load, as far as it can be influenced by our migration method. However, certain constellations of computer hardware can necessitate another distribution of the databases. Therefore the load of the computers is given a low weight for the computation.

Reducing the cost for communication by means of an increased local communication, caused by a corresponding distribution of data, has nearly no weight during the decision for an optimal distribution of data (see 3.1). Thus the concerning parameter has only the lowest weight of all decision-parameters.

Functionality. There are three phases of the distribution function:

1. *Evaluation:* During this phase the information recorded in the profile of accesses is evaluated. Every decision-parameter is, corresponding to its weight, involved in this process in order to gain a predication about the efficiency of the actual distribution of data.
2. *Determination:* The second phase determines a new distribution of data by means of the gained information and of the computed proportions of the first phase. A very efficient method to determine the distribution is to use heuristics which take into account the different decision-parameters and their weights.
3. *Optimization:* The third phase tries to compose the different distributions of data which were found in the second phase due to contradictory decision-parameters. Conflicts between the determined distributions often inhibit the extraction of an optimal distribution of data. In this case it is necessary to deduce a suboptimal solution out of the results of the second phase. This result must fit the given aims as good as possible.

3.5 Cycle-protocol

The cycle-protocol of the migration concept consists of three phases. They work independently of each other and parallel to the operating of the knowledge base.

1. *Protocol-phase :* This phase is used for recording the access history of the different application programs to the existing databases. Moreover, it generates a profile of accesses with detailed information for every access to the databases (see 3.2).
2. *Decision-phase :* The decision-phase is initiated through an event. In this phase the information is evaluated and a new distribution of data is determined. Therefore it uses the distribution function and the information which was gained in the protocol-phase.
3. *Realization-phase :* After the decision-phase the realization-phase starts. During this phase the real physical migration of the data takes place. Therefore the single relations are distributed according to the determined distribution of the decision-phase.

By dividing the cycle-protocol into three phases a sort of pipeline is realized, in which all phases run independently of each other. The second phase uses the information gained in the first phase whereas the first phase already collects new information about the access structure of the different application programs. Then the third phase realizes the data distribution which was computed in the second phase whereas the second phase already evaluates the new information of the first phase. Consequently, the migration mechanism can react quickly on new demands for migration that occur when multiple users and application programs use the knowledge base. Moreover, every component of the migration mechanism is exchangeable without any problems, because there is only a narrow interface between the different phases to exchange their results. Thus, there is no effort to integrate another distribution function or another mechanism to realize the distribution within the migration mechanism.

4 Conclusion and outlook

A basic approach for a migration concept was developed according to the aims of the SFB331 for highly parallel work of the application programs of the knowledge base. This migration concept realizes an optimal distribution of data to the databases of a knowledge base. In order to get a funded migration concept it was developed by the means of an adapted classification scheme for object-oriented systems.

First of all, the aims concerning the migration mechanism were fixed. Especially the aims of a high service-parallelism and an even distribution of database-load must be regarded.

The main part of the migration concept is a distribution function which determines an optimal distribution of data according to the aims. Therefore it uses a profile of accesses which is recorded parallelly to the work of the knowledge base. This profile contains relevant information for the migration mechanism about every access to the databases. Accordingly, several decision-parameters for the distribution function were examined. Every decision-parameter got a certain weight which reflects its importance during the determination of a new data distribution. Furthermore, the functionality of the three phases of the distribution function was presented. First, there is an evaluation of the recorded information which is followed by the determination of a new distribution of data. Finally, an optimization combines the results to an optimal distribution for the actual demands.

In general the migration mechanism is initiated event-related (task-specific). But there also exists the possibility that the user could influence the initiation. This is useful when the user has a-priori knowledge about the expected access structure. Thus the distribution of data could be adapted in advance to this access pattern.

The cycle-protocol consists of three independent phases which run parallelly to the knowledge base. The protocol-phase generates a profile of accesses with detailed information about every access to the databases. During the decision-phase this information is evaluated and a new distribution of data is determined using the distribution function. Finally the realization-phase distributes every single relation according to the determined distribution. By dividing the cycle-protocol into three independent phases a sort of pipeline is realized which guarantees a quick reaction on changing demands in the access pattern and an easy exchange of certain components of the migration mechanism.

Future research will mainly substantiate the concept of the distribution function with regard to the presented decision-parameters. Therefore an explicit mechanism for the evaluation of the profile of accesses and the determination of a new distribution of data must be developed. Certain heuristics adapted to the demands for the migration must be found in order to support the determination and optimization of the decision-process.

If the distribution function is realized, further research will develop strategies that distribute *new* objects in an optimal way. Then optimal access to the databases can be guaranteed from the very beginning.

References

[1] Casavant T.L., Kuhl J.G.: *A Taxonomy of Scheduling in General Purpose Distributed Computing Systems*; IEEE Transaction on Software Engineering, 1988, SE-14 (2), 141-154

[2] Förster C.: *Adaptive Allocation of Computational Requirements to Heterogeneous Networks*; ITG/GI–Fachtagung 'Kommunikation in verteilten Systemen', Stuttgart 1989; Informatik Fachberichte 205, 324–337, Springer Verlag Berlin Heidelberg, 1989

[3] Ghandri K.: *Entwurf, Implementierung und Bewertung eines Konzepts zur parallelen Bearbeitung von Dienstaufrufen in einer realzeitfähigen, verteilten Wissensbasis*; Technische Universität München, Institut für Informatik, Diplomarbeit, Nov. 1993

[4] Hac A.: *Concurrency Control in a Distributed System and Its Performance for File Migration and Process Migration*; -In: Compcon IEEE Computer Society International Conference Digest of Papers, San Francisco, March 1986; Eds.: Bell Alan G.; IEEE, 1986, 426-434

[5] Meyfarth R.: *Demon Concepts in an Active CIM Knowledge Base*; - In: Proceedings of IEEE International Conference 'Robotics and Automation', Cincinnati, Ohio, May 1990, 902-907

[6] Schill A.: *Migrationssteuerung und Konfigurationsverwaltung für verteilte objektorientierte Anwendungen*; Informatik Fachberichte 241, Springer Verlag Berlin Heidelberg, 1990

[7] Schmid O.: *Entwicklung und Realisierung von Konzepten zur Verbesserung der Realzeitfähigkeit einer objektorientierten Wissensbasis für Fertigungsumgebungen*; Technische Universität München , Institut für Informatik, Diplomarbeit, Mai 1994.

[8] Schweiger J.: *Konzepte für eine teamfähige, verteilte Wissensbasis für Fertigungsumgebungen*; Technische Universität München, Institut für Informatik, Technischer Bericht Nr. TUM-I-94-20, Juni 1994

[9] Schweiger J., Ghandri K., Koller A.: *Concepts of a Distributed Real-Time Knowledge Base for Teams of Autonomous Systems*; -In: Proceedings of the IEEE International Conference on Intelligent Robotic Systems and the Real World, Munich, Sept. 12-16, 1994; IEEE Computer Society Press, 1994

[10] Varadarajan R., Ma Y.E.: *An Approximate Load Balancing Model with Resource Migration in Distributed Systems*; -In: International Conference on Parallel Processing (Volume 1: Architecture), August 1988; Eds.: Briggs Faye A.; Pennsylvania: Penn State University, 1988, 13-18