Agents and Databases: A Symbiosis?

Heiko Schuldt

Database and Information Systems Group
Department of Computer Science
University of Basel, Switzerland
heiko.schuldt@unibas.ch

Abstract. Over the last decades, data and information management has been subject to significant changes. Access to data and information is no longer provided by monolithic database systems. Rather, applications need to cope with an increasing number of heterogeneous and distributed data and information sources, ranging from traditional databases, large document collections and information sources on the Internet and the Semantic Web. This also affects the way data and information is searched, accessed, and processed. In particular, the agent community has addressed this change and has spawned the field of *information agents*. An information agent pro-actively searches, retrieves, accesses and maybe even processes information on behalf of its user. Also the database community has faced the challenges stemming from this change by making database functionality available even outside of database systems.

In this paper, we review the recent developments in both fields and show examples of activities which lead to synergies in both communities and which emphasize on the potential for symbiotic co-existence.

Keywords: Cooperative Information Agents, Databases, Hyperdatabase Systems, Agents and Transactions.

1 Introduction

Over the last decades, data and information management has undergone considerable changes. From rather monolithic, database-centric applications where access to data was directly provided by (mostly relational) database management systems (DBMSs), the evolution first led to an increasing number of heterogeneous and distributed data and information sources, ranging from traditional databases and large (multimedia) document collections to information sources on the Internet, and finally to information in the Semantic Web and even to embedded information sources in mobile "smart" objects as they occur in a pervasive computing environment. This development significantly affects the way data and information is searched, accessed, and processed. Both the immense amount of information and the number of different information sources poses a great challenge for appropriate infrastructures for dealing with search, access, management, and processing. In particular, the agent community has addressed this change and has spawned the field of *information agents* [13,9]. A cooperative information agent is a *computational software entity that has access to one or multiple, potentially heterogeneous, and geographically and logically distributed data and information sources, pro-actively acquires, mediates, and maintains relevant information on behalf of its human users or other agents, preferably just-in-time* [4]. Several workshop and conference series have provided the fora necessary for creating a pertinent community and have helped the field in coming of age.

At the same time, database systems have evolved and the database community has also faced the challenges stemming from this change. In particular, databases are more and more hidden behind service interfaces. At the same time, database concepts such as query processing and transactions are provided outside of database systems, at the level of service invocations.

In this paper, we review the recent developments in both fields. In particular, we aim at answering the rather rhetorical question raised in the title of this paper by identifying areas of mutual interest and activities which hopefully lead to synergies in both communities. The hyperdatabase vision [16] will be presented as one example for activities in the intersection between both areas and we describe in more detail two concrete realizatons of this vision to exemplify the relationship between both fields and to stress the possibility for symbiotic co-existence.

The paper is organized as follows: Section 2 briefly introduces the two different fields and identifies the potential for cross-fertilizations. In Section 3, we illustrate the possible symbiosis between information agents and databases by presenting the hyperdatabase vision. In particular, we present two selected hyperdatabase implementations, namely OSIRIS which provides optimized routing of service requests for distributed processes orchestrated by means of cooperating agents and AMOR which provides transactional execution guarantees for cooperating agents. Finally, Section 4 concludes.

2 Information Agents and Databases

In what follows, we first give a brief introduction on information agents and databases, review their development and analyze the relationship between both fields.

2.1 Information Agents

In short, the main task of information agents can be summarized as providing integrated access to information from potentially heterogeneous information sources hosted at several locations. The activities of information agents include the discovery of information sources, the integration of information from different sources and possibly also the processing of information, e.g., to derive new information [13,9].

In more detail, according to the terminology defined by the AgentLink Special Interest Group on Intelligent Information Agents [2], an information agent is a computational software entity that may access one or multiple, distributed and heterogeneous information sources, and pro-actively acquires, mediates, and maintains relevant information on behalf of its user(s) or other agents preferably just-in-time. This includes their ability to semantically broker information by providing a pro-active resource discovery, by mediating between information consumers and providers and finally by offering value-added information services and products to the user or other agents. The latter implies that information agents also act as producers of information and not just facilitators for accessing information, and thus have to take care of the quality of their services. Information agents take over the role of brokers between information sources (e.g., by accessing databases), other agents (or their services, respectively), and human users.

Different information agents may cooperate in order to jointly achieve a common task. Furthermore, they may be subject to dynamic re-configurations, e.g., to react to changes in their environment react or to increase the quality of their service (load balancing, reduction of data transfer, etc.).

2.2 Databases

Relational database systems have been introduced more than thirty years ago. They have been considered as infrastructure and main platform for development of data-intensive applications. The notion of “data independence”, part of Codd’s rules [5], was a breakthrough because programmers were freed from low-level details, e.g., how to access shared data efficiently and correctly, given concurrent access. But already in the late nineties, the prerequisites for application development have changed dramatically. Storage and communication has become fairly cheap, and the internet has started to dominate modern information infrastructures. Consequently, the role of database concepts had to be re-visited and newly determined. Undoubtedly, the database system has played and still plays an important role. However, it has more and more degenerated to a storage manager, far away from the applications. In this situation, about a decade ago, researchers started to question the role of databases for future distributed information systems engineering (“Databases Breaking out of the Box” [25] and the Lowell Database Research Self-Assessment Report [1]).

One of the consequences of this development is that databases are increasingly becoming invisible, although they still constitute the backend-tier of data-intensive applications. Rather, data management and thus databases are hidden behind service interfaces. Thus, data-intensive applications have to deal with services description and registration instead of relational schema definition, service instances instead of relations, or service invocations instead of relational operators for accessing or manipulating data. Database research thus more and more needs to cope with database functionality at higher levels of semantics, e.g., optimal routing of service requests as opposed to query optimization, or transactional execution guarantees for composite services to just name a few.

2.3 Symbionts or Predators, Peaceful Coexistence or Mutual Indifference?

In light of these developments, it is obvious that the relationship between information agents and databases cannot be characterized by just mutual indifference.

It even significantly goes beyond a peaceful coexistence since information agents are more than only clients to databases.

From the point of view of information agents, databases are still the resource managers which persistently store information. But even more, the broader notion of databases provides the necessary protocols and mechanisms for coordinating agent interactions, e.g., for providing provably correct and transactionally safe multi-agent executions or for routing service requests in an optimal way.

At the same time, higher level database functionality can significantly benefit from advances in the area of information agents. This includes the ability to semantically integrate information from different sources, to negotiate quality of service for interactions, to proactively discover resources, or to dynamically adapt to changing environments.

Apparently, there are many potential synergies between both fields. Some of them are addressed in recent initiatives and projects which are stemming from the database community and which aim at making database functionality available outside of databases. Thus, these projects implicitly address issues which are of high practical impact also for information agent interactions. Hyperdatabases [15,16,17], InfoSphere [14], or AutoGlobe/ServiceGlobe [6] are some examples out of a longer list of similar initiatives which hopefully lead to a sustainably symbiotic relationship between both fields.

In what follows, we will present in detail the hyperdatabase project which has originated at ETH Zurich and which is now being continued at the University of Basel. In particular, we briefly present the underlying hyperdatabase vision for the management of future information spaces and we will present two implementations of the hyperdatabase vision which provide database functionality at higher level of abstractions, outside of a database, which is supposed to also facilitate and ameliorate distributed agent-based applications.

3 Hyperdatabases

This section first briefly introduces the hyperdatabase vision and then presents two implementations of this vision for which we believe they will have a strong impact on the way distributed information agents interact. A more detailed summary of the hyperdatabase vision and its realizations can be found in a recent survey paper [16].

3.1 The Hyperdatabase Vision

The driving forces behind the hyperdatabase vision are mainly based on two observations. First, that the volume of data and information is significantly increasing and undergoes continuous changes while being inherently distributed and heterogeneous. Second, non-relational data sources such as, for instance, image, video, audio collections are increasingly gaining importance. Thus, a holistic approach to managing the "information space" of the future, i.e., the universe of all information sources, needs a radical departure from the traditional database thinking by moving up to a much higher level of abstraction. In short, a hyperdatabase administers objects that are composed of objects and transactions that are composed of transactions. Thus, it provides database functionality not only over many distributed databases but in a more general way on top of distributed components and services with various functionality in a networked environment.

With hyperdatabases, the notion of data independence is generalized in the form of "higher order data independence". This includes the immunity of application programs not only against changes in storage and access structure, but also against changes in location, implementation, workload, the number of replica of software components and their services. The relation between databases and hyperdatabases can be briefly characterized as follows: a database is a platform for clients concurrently accessing shared data which needs data definition, data manipulation, and transactions at the interface. Internally, the database management system performs query optimization, provides correctness for parallel access, recovery, persistence, load balancing, and guarantees a high degree of availability. Similarly, a hyperdatabase is a platform for clients, concurrently accessing shared application services; thus, as opposed to shared data in a database, it has to deal with shared components and services. At the interface, a hyperdatabase has to provide component and service definition and description, service customization, transactional processes encompassing multiple service invocations. Internally, the hyperdatabase performs optimization of client requests, routing, scheduling, and parallelization, correctness of concurrent accesses, flexible failure treatment, providing guaranteed termination (i.e., a generalized form of atomicity), availability, flexible recovery, and scalability. Table 1 summarizes the analogy.

Most importantly and in contrast to traditional database technology, a hyperdatabase infrastructure must not follow monolithic system architecture but must be fully distributed over all participating nodes in a network. Every node is equipped with an additional thin software layer, a so-called hyperdatabase layer (which, in the terminology of the agent community, is actually an information agent). Each deployment of the hyperdatabase layer can be considered as an agent which offers dedicated services and/or provides access to information. Thus, one of the main challenges of hyperdatabases is the communication and coordination of these hyperdatabase layers.

3.2 Hyperdatabase Projects

The following two sections present in more detail two concrete implementations which arose from the hyperdatabase vision. Both address the distributed retrieval and/or processing of information by a set of cooperating hyperdatabase layers (agents). The first focuses on distributed, process-based applications while the second addresses transactional semantics in these distributed settings.

OSIRIS: A Hyperdatabase Implementation for Distributed Process-based Applications. The proliferation of service-oriented computing and in particular of (Web) services had a strong impact on information systems. System

Table 1. Analogy between DBMSs and the Hyperdatabases (from [16])

Database Management	**Hyperdatabase Infrastructure**
Relational schema definition	Service definition and registration
Relational schema extension	New service registration
Relation	Service instance
Access to relation	Service invocation
Query and update language	Process definition language
Transaction	Transactional process
ACID Guarantees	Correct execution and guaranteed termination of process
Undo operation	Inverse service invocation
Redo operation	Repeatable service invocation
Indexing	Feature extraction and feature space organization
Query Optimization	Optimal process routing
Physical Database Design	Configuration Design by service allocation and replication

support for the invocation of single services is widely available, due to standardized protocols and formats (e.g., WSDL and SOAP). Beyond these basics, the most important challenges are the management of existing services and their evolution, the composition of existing services into a coherent whole by means of processes, and the optimization of service requests to guarantee a high degree of scalability in order to deal with an increasing number of services, processes, and users.

OSIRIS (Open Service Infrastructure for Reliable and Integrated process Support) [22,23] is a novel infrastructure for distributed service-oriented applications. It primarily focuses on the scalable and reliable execution of composite services, also called processes. OSIRIS provides basic mechanisms which can also be applied to distributed, cooperating information agents in jointly achieving a common task. Process-based applications are either explicitly specified, according to the paradigm of programming in the large [26], or are individually and automatically created by a dedicated planner (e.g., [10,11,24]).

OSIRIS consists of a set of agents (so-called hyperdatabase layers). These agents interact in a decentralized, peer-to-peer style for executing processes [20]. In addition, OSIRIS considers several global repositories. While only the agents are responsible for process execution, the global repositories collect metadata on the overall system and apply sophisticated replication mechanisms (based on publish/subscribe techniques) for control flow dependencies from the global repositories to the agents. At run-time, this guarantees that no single point of failure is involved in the execution of processes and allows to provide sophisticated load balancing strategies (selection of the least loaded peer which provides a dedicated service). For this, the concrete service binding is determined at run-time depending on the load of agents and costs of invoking a particular service instance [21].

In order to minimize information exchange between repositories and agents, only a minimal set of information is replicated, i.e., only the information an agent needs to drive the execution of those process instances that it might potentially be responsible for (for which it provides services). Among all global OSIRIS services, the most important ones for distributed and decentralized process execution are the the process repository which holds the global definitions of all processes types, the service registry which is a directory of all available services in the system provided by OSIRIS agents, and the load repository which manages information on the load of all agents in the system.

OSIRIS' decentralized and distributed approach to process execution is illustrated in Figure 1. Different service types are depicted with different shapes, execution orders are illustrated by directed edges. Agents (OSIRIS layers) are sitting on top of all service providers and allow them to make available their services to OSIRIS processes. In the center, some of the core OSIRIS services are displayed. Figure 1 also shows how a process description is replicated in small pieces to the OSIRIS agents (dotted lines). Finally, after replication, enough process and service meta information is locally available to allow for peer-to-peer process execution. In particular, when a process is instantiated and executed, the OSIRIS agents can decide on their own, based on locally replicated information, where to route a request to (solid lines between OSIRIS agents). This makes sure that process execution takes place in a decentralized and distributed way and guarantees a high degree of scalability.

Distributed Concurrency Control for Processes. In databases, transactional execution guarantees are of high importance and have strong practical impact in a large number of applications. In general, this is also true for hy-

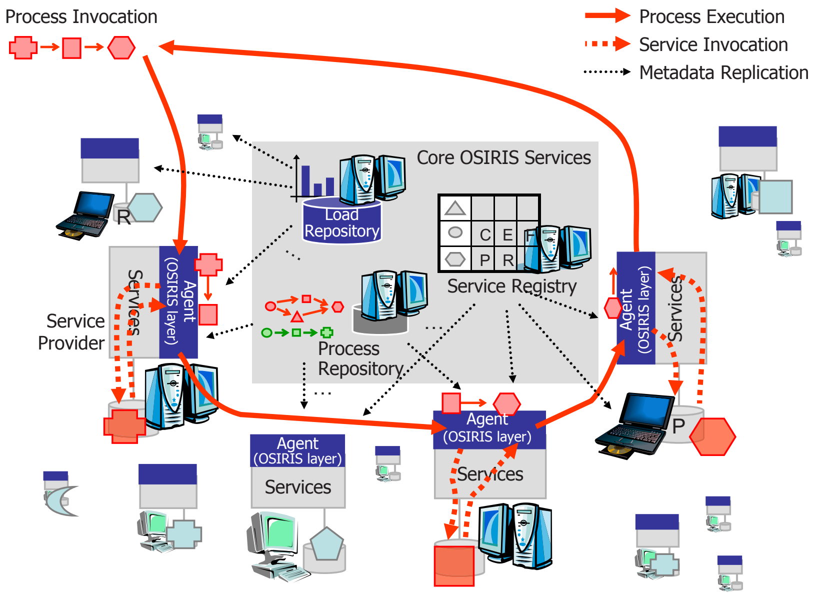

Fig. 1. Distributed Agent-based Execution of OSIRIS Processes

perdatabases. However, isolated and atomic behavior for concurrent distributed, service-based applications needs to take into account the higher level semantics of services (compared to rather low-level database operations). Transactional processes [19] consider these constraints and provide process support with transactional guarantees over distributed components using existing services as a generalization of traditional database transactions. Essentially, transactional processes exploit the termination semantics of the individual services they contain. Each service is either *compensatable*, *retriable*, or *pivot*, following the model of flexible transactions [27]. The effects of compensatable services can be semantically undone after the invocation has successfully returned. Retriable services are guaranteed to terminate correctly, even if they have to be invoked repeatedly. In this case, the last invocation succeeds while all previous invocations of this service do not leave any effects. [18] presents a more advanced distinction between termination classes, based on execution costs of services. Pivot services are those that cannot be compensated, due to the lack of an inverse service, or which are not appropriate for compensation due to their high costs.

On the basis of the transactional process model, the AMOR (Agents, MObility and tRansactions) approach [8,7] allows to provide global transactional guarantees, i.e., atomicity and isolation applied at the level of processes without any global component involved. Conventionally, isolation and atomicity are enforced using a locking protocol like the strict two-phase locking (2PL) in combination with a global commit protocol like the two-phase commit (2PC) [12] and require a centralized coordinator. These protocols are not applicable in completely distributed agent-based applications. AMOR uses a novel protocol which is based on decentralized serialization graph testing to ensure global correctness (concurrency control and recovery) in peer-to-peer environments without a global coordinator. Essentially, each agent is equipped with partial knowledge (local serialization graph containing information on conflicts with other agents) that allows them to coordinate. Globally correct execution is achieved by communication among dependent agents and can even be enforced in case of incomplete local knowledge. It can be guaranteed that each agent can decide at commit time whether it is able to safely commit its work or whether it has to wait on other agents to commit their work first before they can proceed.

Thus, AMOR provides the basic protocol that can be used to add transactional semantics to any kind of agent cooperation without imposing a dedicated infrastructure for this coordination, similarly to the way it has brought forward P2P systems with transactional semantics [3].

4 Conclusion

The significant growth of information over the last years has led to an increasingly large number of information sources in various formats and at different locations. This information might even be subject to frequent changes and complex interdependencies. In order to support applications in dealing with this wealth of heterogeneous information, novel approaches are required to access these information sources, to mediate between them and to provide value-added services. These problems are in the focus of information agents, which take over these tasks on behalf of their users or other agents. Most importantly, tasks are usually delegated to groups of (possibly specialized) agents which solve them in a collaborative way.

At the same time, this evolution has also led to a re-thinking of database research. Databases are no longer in the center of applications but are hidden to the applications behind service interfaces. Nevertheless, well known guarantees from databases like optimized access and transactional execution are still needed, but at a higher level of semantics, namely the invocation of services.

The activities of the information agent and database communities have led to highly complementary results and research activities are more and more directed towards bringing both fields closer together. So the rather rhetorical question from the title of this paper can be clearly answered: there is indeed a high potential for synergies and cross-fertilization between both fields which is visible in the promising results of some initiatives, and the research agendas will hopefully be much closer aligned in the near future.

In this paper, we have reported on some activities which originated from the database community and which, as we believe, will also have a strong impact also on cooperative information agents. In particular, we have summarized the hyperdatabase vision which aims at applying database system concepts outside of databases. In addition, with OSIRIS and AMOR, we have presented two hyperdatabase implementations. The first aims at providing optimized routing of service requests, as a generalization of query optimization in databases, while the latter focuses on providing transactional execution guarantees for composite services, as a generalization and extension of ACID guarantees known from database transactions.

Over the last ten years, the hyperdatabase vision has been implemented, exploited, and evaluated in a large variety of applications. However, the hyperdatabase vision is not static but needs to evolve with the ongoing advancements and new trends in large-scale, distributed and heterogeneous information spaces. Current activities in the context of the hyperdatabase vision, for instance, consider additional support for context-aware service composition and semantic failure handling. Essentially, when considering the current context (e.g., location) of a user or her individual preferences, personalized process-based applications can be either newly created or existing ones can be automatically adapted. This includes the customizaiton and generation of processes using semantic Web services, their reliable distributed execution, and finally the exploitation of semantics for failure handling purposes – the latter will significantly benefit from recent work done in the context of cooperative information agents.

References

1. Abiteboul, S., Agrawal, R., Bernstein, P.A., et al.: The Lowell Database Research Self-Assessment. Communications of the ACM 48(5), 111–118 (2005)
2. AgentLink. Special Interest Group on Intelligent Information Agents, `http://www.dbgroup.unimo.it/IIA/`

3. Antony, S., Agrawal, D., Abbadi, A.E.: P2P Systems with Transactional Semantics. In: Proceedings of the 11th International Conference on Extending Database Technology (EDBT 2008), Nantes, France, March 2008, pp. 4–15. ACM Press, New York (2008)
4. CIA. International Workshop Series on Cooperative Information Agents, `http://www-ags.dfki.uni-sb.de/~klusch/IWS-CIA-home.html`
5. Codd, E.F.: The Capabilities of Relational Database Management Systems. IBM Research Report, San Jose, California, RJ3132 (1981)
6. Gmach, D., Krompass, S., Scholz, A., Wimmer, M., Kemper, A.: Adaptive Quality of Service Management for Enterprise Services. ACM Transactions on the Web (TWEB) 2(1) (Febuary 2008)
7. Haller, K., Schuldt, H., Schek, H.-J.: Transactional Peer-to-Peer Information Processing: The AMOR Approach. In: Chen, M.-S., Chrysanthis, P.K., Sloman, M., Zaslavsky, A. (eds.) MDM 2003. LNCS, vol. 2574, pp. 356–361. Springer, Heidelberg (2003)
8. Haller, K., Schuldt, H., Türker, C.: Decentralized Coordination of Transactional Processes in Peer-to-Peer Environments. In: Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management, Bremen, Germany, pp. 28–35. ACM Press, New York (2005)
9. Klusch, M. (ed.): Intelligent Information Agents. Springer, Heidelberg (1999)
10. Lopes, A., Costa, P., Bergenti, F., Klusch, M., Blankenburg, B., Möller, T., Schuldt, H.: Context-aware Secure Service Composition Planning and Execution on E-Health Environments. In: Proceedings of the European Conference on eHealth (ECEH 2006), Fribourg, Switzerland, pp. 179–190 (October 2006)
11. Möller, T., Schuldt, H., Gerber, A., Klusch, M.: Next Generation Applications in Healthcare Digital Libraries using Semantic Service Composition and Coordination. Health Informatics Journal (HIJ), Special Issue on Health Digital Libraries 12, 107–119 (2006)
12. Özsu, M.T., Valduriez, P.: Principles of Distributed Database Systems, 2nd edn. Prentice Hall, Englewood Cliffs (1999)
13. Papazoglou, M.P., Laufmann, S.C., Sellis, T.K.: An Organizational Framework for Cooperating Intelligent Information Systems. International Journal on Cooperative Information Systems 1(1), 169–202 (1992)
14. Pu, C., Schwan, K., Walpole, J.: Infosphere Project: System Support for Information Flow Applications. SIGMOD Record 30(1), 25–34 (2001)
15. Schek, H.-J., Böhm, K., Grabs, T., Röhm, U., Schuldt, H., Weber, R.: Hyperdatabases. In: Proceedings of the First International Conference on Web Information Systems Engineering (WISE 2000), Hong Kong, China, June 2000, pp. 14–25. IEEE Computer Society, Los Alamitos (2000)
16. Schek, H.-J., Schuldt, H.: The Hyperdatabase Project – From the Vision to Realizations. In: Proceedings of the 25th British National Conference on Databases (BNCOD 25), Cardiff, UK, July 2008. LNCS, vol. 5071. Springer, Heidelberg (2008)
17. Schek, H.-J., Schuldt, H., Weber, R.: Hyperdatabases: Infrastructure for the Information Space. In: Proceedings of the Sixth IFIP Working Conference on Visual Database Systems (VDB 2002), Bisbane, Australia, May 2002, pp. 1–15. Kluwer Academic Publishers, Dordrecht (2002)
18. Schuldt, H.: Process Locking: A Protocol based on Ordered Shared Locks for the Execution of Transactional Processes. In: Proceedings of the 20th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2001), Santa Barbara, CA, USA, May 2001, ACM Press, New York (2001)

19. Schuldt, H., Alonso, G., Beeri, C., Schek, H.-J.: Atomicity and Isolation for Transactional Processes. ACM Transactions of Database Systems (TODS) 27(1), 63–116 (2002)
20. Schuler, C., Schuldt, H., Türker, C., Weber, R., Schek, H.-J.: Peer-to-peer Execution of (Transactional) Processes. International Journal on Cooperative Information Systems 14(4), 377–406 (2005)
21. Schuler, C., Türker, C., Schek, H.-J., Weber, R., S.H.: Scalable Peer-to-Peer Process Management. International Journal of Business Process Integration and Management (IJBPIM) 1(2), 129–142 (2006)
22. Schuler, C., Weber, R., Schuldt, H., Schek, H.-J.: Peer-to-Peer Process Execution with OSIRIS. In: Orlowska, M.E., Weerawarana, S., Papazoglou, M.P., Yang, J. (eds.) ICSOC 2003. LNCS, vol. 2910, pp. 483–498. Springer, Heidelberg (2003)
23. Schuler, C., Weber, R., Schuldt, H., Schek, H.-J.: Scalable Peer-to-Peer Process Management – The OSIRIS Approach. In: Proceedings of the IEEE International Conference on Web Services (ICWS 2004), San Diego, CA, USA, June 2004, pp. 26–34. IEEE Computer Society Press, Los Alamitos (2004)
24. Schumacher, M., Helin, H., Schuldt, H. (eds.): CASCOM: Intelligent Service Coordination in the Semantic Web. Whitestein (2008)
25. Silberschatz, A., Zdonik, S.B.: Database Systems - Breaking Out of the Box. SIGMOD Record 26(3), 36–50 (1997)
26. Wiederhold, G., Wegner, P., Ceri, S.: Toward Megaprogramming. Commununications of the ACM 35(11) (1992)
27. Zhang, A., Nodine, M.H., Bhargava, B.K.: Global Scheduling for Flexible Transactions in Heterogeneous Distributed Database Systems. IEEE Transactions on Knowledge and Data Engineering (TKDE) 13(3), 439–450 (2001)