Data Management Research at Technische Universität Darmstadt

A. Buchmann and M. Cilia
Databases and Distributed Systems Group, Department of Computer Science
Technische Universität Darmstadt - Darmstadt, Germany
<lastname>@dvs1.informatik.tu-darmstadt.de

Abstract

The Databases and Distributed Systems Group at Technische Universität Darmstadt is devoted to research in the areas of data management middleware and reactive, event-based systems. Special emphasis is placed on handling the flow of data and events in a variety of environments: publish/subscribe mechanisms, information dissemination and integration, ubiquitous computing, peer-to-peer infrastructures, and a variety of sensor-based systems ranging from passive RFID infrastructures to active wireless sensor networks. A special concern is placed on non-functional aspects of the middleware, such as performance, scalability and security, where members of our group are involved in the definition of the SPEC family of benchmarks for J2EE (SPECjAppServer200x) and JMS.

1 Introduction

The work in the group is based on several observations derived from the convergence of technologies:

- the deployment of smart devices requires the continuous monitoring of events and context data for their correct interpretation;
- the miniaturization of sensors and their ubiquitous deployment will result in massive amounts of sensor data that must be processed, often under real-time constraints;
- the heterogeneity of resources of the participating nodes and their unstable connectivity leads to mixed-mode systems with special needs with respect to consistency, availability, security, etc.;
- huge distributed systems must be capable of detecting and correcting failures and return autonomously to stable operation;
- new business strategies, such as event-driven supply chain management and zero-latency enterprises, depend on the timely dissemination of information and business events.

The basic premise underlying our work is that information flows and is no longer confined to repositories. Therefore, traditional pull-based access mechanisms to stagnant data are no longer sufficient. In addition, a reliable infrastructure for management of streams of data and events is needed. The importance of this infrastructure will increase as we move to a world populated by huge amounts of interconnected devices with different capabilities that will react and automate processes on our behalf.

This research survey is organized into 3 sections: Data Dissemination, Peer-to-Peer meets Pub/Sub, and Performance modeling.

2 Data Dissemination

Data dissemination is one of the core problems in monitoring applications ranging from RFID-based logistics and warehouse control to ambient intelligence. We have developed solutions based on the Publish/Subscribe paradigm and have extended the basic content-based routing to accommodate heterogeneity, and developed composition mechanisms for subscriptions (filters) and notifications (data and events). We have addressed quality of service and management issues through the development of middleware-mediated transactions and scopes. We provided for reactive capability that is modular and composable at the subscriber end, and are currently extending the publish/subscribe notification system to accommodate mobility [14], deal with context information [29], and provide some basic security [13].

2.1 Routing strategies

We developed a notification service framework called REBECA. It basically offers a distributed event notification service to which applications and other system services are connected as clients. These clients act as producers and consumers of notifications. The notification service itself is an overlay network in the underlying system, consisting of a subset of nodes connected in a network of event brokers. The brokers receive notifications, filter and forward them in order to deliver published notifications to all attached consumers having a matching subscription.

We carried out several experiments on top of REBECA. They show that in large-scale systems, more advanced content-based routing algorithms must be applied [23]. Those algorithms exploit commonalities among subscriptions in order to reduce routing table sizes and message overhead. We have investigated three of them, identity-based routing, covering-based routing [6], and merging-based routing [22]. Identity-based routing avoids forwarding of subscriptions that match identical sets of notifications. Covering-based routing avoids forwarding of those subscriptions that only accept a subset of notifications matched by a formerly forwarded subscription. Note that this implies that it might be necessary to forward some of the covered subscriptions along with the unsubscription if a subscription is cancelled. Merging-based routing goes even further. In this case, each broker can merge existing routing entries to a broader subscription, i.e., the broker creates new covers.

Advertisements can be used as an additional mechanism to further optimize content-based routing. They are filters that are issued (and cancelled) by producers to indicate (and revoke) their intention to publish certain kinds of notifications. If advertisements are used, it is sufficient to forward subscriptions only into those subnets of the broker network in which a producer has issued an overlapping advertisement, i.e., where matching notifications can be produced. Advertisements can be combined with all routing algorithms discussed above.

2.2 Filters and composition

Events can be either primitive or composite. In most practical situations primitive events, e.g. events detected by basic sensors or produced by applications, must be combined. Usually, this composition or aggregation relies on an event algebra that may include operators for sequence, disjunction, conjunction, etc.

However, these event algebras and consumption policies depend on a total order of events and are based on point-based timestamps of a single central clock. These assumptions are invalidated by the inherent characteristics of distributed environments. Therefore, the event occurrence time must be considered to be indeterminate to some extent. As a consequence, time indeterminacy must be reflected in the time model and explicitly recognized and reported when composing events in distributed and heterogenous systems [19].

We have built an event aggregation service that is based on the principles of components and containers [9]. Containers control the event aggregation process while components define the event operators logic. As mentioned before, the aggregation service is treated like any other event consumer that can subscribe to events, it aggregates them and finally publishes the aggregated event. The handling of time indeterminacy and network delays are encapsulated in such a container.

Additionally, in many cases the traffic of messages within a notification service can be reduced by applying filters. For this purpose a framework for filter definition is under construction. These filters can simply discard events according to some pattern (one in ten), or by placing them close to their source where a straightforward analysis of relevant changes (for instance, the analysis of regular events signaled by a temperature sensor) is carried out.

2.3 Data integration

In a realistic environment events are produced at heterogeneous/diverse sources. These events encapsulate data, which can only be properly interpreted when sufficient context information about its intended meaning is known. In general, this information is left implicit and as a consequence, it is lost when data/events are exchanged across institutional or system boundaries. Combining or interpreting data from different sources leads inevitably to problems if the meaning of terms is not shared.

In the context of notification services, consumers need to know about the content of the events/messages that are being exchanged in order to express their subscriptions. That means, that consumers must know details about the representation and assumed semantics of message content. Today, notification services do not support this leaving required information about data semantics implicit. Without this information event producers and consumers are expected to fully comply with implicit assumptions made by participants.

The approach we have taken solves this problem by providing a concept-based layer on top of the delivery mechanism [7]. This layer provides a higher level of abstraction in order to express subscription patterns and to publish events with the necessary information to support their correct interpretation outside the producers' boundaries. This was achieved by relying on the MIX model [3, 4] for the representation of data (i.e., event content). MIX directly supports data integration by making the concept of semantic context (i.e., the explicit description of implicit assumptions about the meaning of the data) and conversion functions (which allow the automatic conversion of data/events from different sources to a common context) first class citizens of the model itself.

2.4 Multi-hop transaction support

In distributed settings, the application process typically spans multiple transactional information systems. Grouping the information access into a single distributed transaction requires resources to be locked for the duration of the transaction and termination must be coordinated by a 2-phase-commit protocol. While this approach is realized in standardized and commonly applied middleware services [24], the applicability thereof is restricted to tightly coupled systems and thus is not suitable for the integration of autonomous components.

In the event-based architectural style the event producer is decoupled from the event consumer through the mediator. Therefore, any transaction concept in an event-based system must include the mediator. On the other hand, applications will be implemented in some (object-oriented) programming language. The challenge is therefore, to combine notifications with conventional transactional object requests into middleware mediated transactions (MMT) [21]. MMTs extend the atomicity sphere of transactional object requests to include mediators and/or final recipients of notifications.

In order to integrate producers, mediators and subscribers, a more flexible transactional framework was developed [20]. This framework provides the means to couple the visibility of event notifications to the boundaries of transaction spheres and the success or failure of (parts of) a transaction. It also describes the transactional context in which the consumer should execute its actions. It specifies the dependencies between the triggering and the triggered transactions, dynamically spanning a tree of interdependent transactional activities.

2.5 Reactive capabilities

Emerging trends like, event-driven supply chain management, the zero-latency enterprise, or ambient intelligence applications depend on the timely dissemination of data but also on the proper reaction to those events. The Event-Condition-Action rule (ECA-rule) approach fits very well in this context, but does not always require a full-fledged database support. We decomposed the traditional processing of ECA-rules (typically embedded in active databases) into its elementary and autonomous parts [8]. These parts are responsible for event aggregation (see Section 2.2), condition evaluation and action execution. The processing of rules is then realized as a composition of these elementary services on a per rule basis. This composition forms a chain of services that are in charge of processing the rule in question. These elementary services interact among them based on the notification service. As mentioned before, the reactive service is treated like any other event consumer that can subscribe to events. When events of interest (i.e. those that trigger rules) are notified, the corresponding rule processing chain is automatically activated. Elementary services (i.e. action execution) that interact with external systems or services use *plug-ins* for this purpose. Besides that, plug-ins are responsible for maintaining the semantic target context of the system they interact with making possible the meaningful exchange of data. This service is used in the context of online meta-auctions [5], the Internet-enabled car [10] and an RFID-based supply chain scenario as well.

2.6 Scoping

Despite the numerous advantages offered by the loose coupling of event-based interaction, a number of drawbacks arise from the new degrees of freedom. Event systems are characterized by a flat design space in which subscriptions are matched against all published notifications without discriminating producers. This makes event systems difficult to manage. A generic mechanism is needed to control the visibility of events, e.g. for security reasons, and for structuring sets of producers and consumers, extending visibility beyond the transactional aspects (as presented early in Section 2.4). Operational controls and management tasks can then be bound to these structures.

Scopes [12] allow system engineers to exert explicit control on the event-based interaction; it is a functionality orthogonal to the different layers of a notification service. We see scopes as the means by which system administrators and application developers can configure an event-based system. Scopes offer an abstraction to identify structure and to bind organization and control of routing algorithms, heterogeneity support, and transactional behavior to the application structure. They delimit application functionality and contexts, controlling side effects and associating ontologies at well-defined points in the system. This is of particular importance as platforms of the future must be configurable not only at deployment time but also once an application is in operation. We have introduced scopes in two different environments such as the J2EE platform [11] and Wireless Sensor Networks [27, 26].

3 Peer-to-Peer meets Pub/Sub

The convergence of technologies we alluded to in the introduction and our interest in non-functional properties, such as scalability and robustness, also pushed us to look at the question: what happens if you try to combine the behaviour of a publish/subscribe system with the resilience of a peer-to-peer substrate. This question does not only have academic appeal as can be seen from the gaming scenario we use both as a motivation and for requirements.

3.1 Gaming as a motivation

The gaming industry is about to surpass the movie industry in total revenue. One of the fastest growing branches is the one of massive multi-player online games. MMOGs muster followings of several hundred thousands of players who subscribe on a monthly basis to play over years in a virtual world divided into shards. In the current client/server architectures, technical limitations impose a limit of about 7 000 players per game server. One of the huge intangibles when launching such a game is the success rate. If the success rate is estimated too optimistically, a huge investment in infrastructure is wasted; on the contrary, a pessimistic estimate may lead to sluggish performance and the loss of favour in the gaming community. An interesting solution is to develop MMOGs on a P2P infrastructure. This idea exploits the fact that gamers tend to have state of the art hardware and communications. Migrating MMOGs to a P2P platform implies pushing game events to many servers under controlled conditions and raises many quality of service issues: latency, robustness, scalability, consistency of the game states, security, etc. We have been looking at many of these issues from the perspective of how to control cheating in a gaming environment without central controls [15].

3.2 Building P2P networks with controlled QoS

One of the biggest challenges in the P2P community is to build systems with controlled quality of service. In most cases, P2P systems are laboriously handcrafted and a posteriori their behaviour might be studied. We have approached the building of QoS aware P2P systems from a database point of view [2]. In a first step, nodes with certain quality attributes (e.g. 90% availability) are declaratively selected from a node database. In a second step, a parameterized topology is selected, according to which the nodes will be connected. This tool allows us to configure new P2P networks with different quality attributes and topologies with a few lines of code [1]. While the present system represents progress in the right direction and allows us to easily build P2P systems with nodes of individual QoS characteristics, it is still a long way to predicting global QoS, which is the subject of ongoing work.

3.3 The Rendezvous problem

In many distributed applications, pairs of queries and values are evaluated by participating nodes. Examples include keyword searches for documents, selection queries on tuples, or matching of filters and notifications in publish/subscribe systems. In a distributed system the key question is: where should the evaluation take place and how can data movement be minimized. Work on this generic problem in the P2P context resulted in the bit-zipper approach [28], which deduces from the coding scheme, at which node of a distributed system query and data (or filter and notifications) should optimally meet. The bit-zipper is provenly optimal (in terms of messages sent) for problems in which all pairs of queries and data must be evaluated. Where flooding to N nodes was previously the only fall-back, the bit-zipper needs only $O(\sqrt{N})$. Ongoing research is looking at lower and upper bounds and further generalizations in the processing of queries in P2P systems.

4 Performance Modeling, Analysis and Prediction

Modern applications are typically built on highly distributed, multitiered platforms that are deployed in heterogeneous environments. The complexity of such systems makes it difficult to anticipate the performance of a given deployment. Load testing and benchmarking is quite useful for the identification of bottlenecks, however, it is impractical and costly since a deployment environment like the final application deployment is required. For any kind of extrapolation or decisions earlier in the design cycle, performance models are needed. The Databases and Distributed Systems Group is active in both areas, for traditional enterprise applications as well as notification services and event-based systems.

4.1 Enterprise applications

As members of the SPEC Java Subcommittee we have been actively involved in developing the SPECjAppServer family of benchmarks for the J2EE platform [25]. This has also allowed us to calibrate and validate our performance models against large-scale deployments of the benchmarks [16, 17].

The performance models that were developed in the group are based on traditional queuing networks [16] and on queuing Petri nets [17]. The QN models are quite accurate for modelling throughput, however, they suffer from certain drawbacks when modelling response time. QN models are suitable for modelling active resources, such as CPUs, but are inadequate to model software contention, as they do not provide any means for modelling synchronization. This problem can be solved by using QPN-based models. QPNs insert queues in the places of a Petri net and provide a good tool for modelling synchronization. However, they suffer from the common problem of state space explosion.

The solution was the development of a simulator based on QPNs. This simulator has been calibrated against analytical models where possible and against a wide range of deployments of the SPECjAppServer2004 benchmark. These deployments include both commercial as well as open source J2EE platforms on individual servers as well as clusters [18].

4.2 Asynchronous interactions

Current performance work is centered on the development of benchmarks and performance models for asynchronous interactions. Members of the group are currently involved in the development of a benchmark for JMS in the context of the SPEC Java Subcommittee. In other cooperation with industry, we evaluated the performance of SAP's AutoID infrastructure. Through these activities we are in a position to experiment with some industry strength RFID deployment scenarios.

Recently we have been making progress on the development of load generation tools for event-based systems. Part of this effort is the visualization of the behaviour of the analyzed platforms as a whole and through introspection of individual components. The latter is achieved through the application of Aspect Oriented Programming techniques that allow us to monitor the internal operation of individual components.

The long-term goal is to develop both analytic models and simulators for event-based asynchronous interactions and calibrate and validate them against a large application scenarios reflected in an industrial strength benchmark.

Acknowledgements

We would like to acknowledge the contributions of the many present and past members of the Databases and Distributed Systems Group who helped shape the present research and whose results we summarized: Jose Antolini, Mario Antolini, Stefan Behnel, Christof Bornhövd, Silvana D'Cristofaro, Ludger Fiege, Cristian Fiorentino, Felix Freiling, Pablo Guerrero, Alejandro Houspanousian, Kai Juse, Dimka Karastoyanova, Roger Kilian-Kehr, Christoph Liebig, Matthias Meixner, Gero Mühl, Diego Palmisano, Kai Sachs, Sebastian Salvucci, Jan Stefan, Wesley Terpstra, Björn Weis, Andreas Zeidler. Thanks go also to previous doctoral students who helped set the stage and the many master students who made their contributions and life so enjoyable.

References

[1] S. Behnel and A. Buchmann. Models and languages for overlay networks. In *Proc. of VLDB Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2005)* , Trondheim, Norway, August 2005.

[2] S. Behnel and A. Buchmann. Overlay networks - implementation by specification. In *Proc. of Middleware 2005*, Grenoble, France, November 2005. (To appear).

[3] C. Bornhövd. *Semantic Metadata for the Integration of Heterogeneous Internet Data (in German)*. PhD thesis, Darmstadt University of Technology, 2000.

[4] C. Bornhövd and A.P. Buchmann. A Prototype for Metadata-Based Integration of Internet Sources. In *Proc. of CAiSE*, volume 1626 of *LNCS*, 1999.

[5] C. Bornhövd, M. Cilia, C. Liebig, and A.P Buchmann. An Infrastructure for Meta-Auctions. In *Proc. of WECWIS'00*, June 2000.

[6] Antonio Carzaniga, David S. Rosenblum, and Alexander L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19(3):332–383, 2001.

[7] M. Cilia, M. Antollini, C. Bornhvd, and A. Buchmann. Dealing with heterogeneous data in pub/sub systems: The Concept-Based approach. In *Intl Workshop on Distributed Event-Based Systems (DEBS'04)*, May 2004.

[8] M. Cilia, C. Bornhövd, and A. P. Buchmann. Moving Active functionality from Centralized to Open Distributed Heterogeneous Environments. In *Proc. of CoopIS'01*, volume 2172 of *LNCS*, 2001.

[9] M. Cilia, C. Bornhvd, and A. Buchmann. CREAM: an infrastructure for distributed, heterogeneous event-based applications. In *Proc. of CoopIS'03*, volume 2888 of *LNCS*, pages 482–502, Catania, Italy, November 2003. Springer.

[10] M. Cilia, P. Hasselmeyer, and A.P. Buchmann. Profiling and Internet Connectivity in Automotive Environments. In *Proc. of VLDB'02*, August 2002.

[11] Dan Dobre. A framework for engineering J2EE-based publish/subscribe applications with scopes. Master's thesis, Technische Universitt Darmstadt, Dept of Computer Science, Germany, 2004.

[12] L. Fiege, M. Mezini, G. Mühl, and A.P. Buchmann. Engineering event-based systems with scopes. In *Proc. of ECOOP'02*, volume 2374 of *LNCS*, 2002.

[13] L. Fiege, A. Zeidler, A. Buchmann, R. Kilian-Kehr, and G. Mhl. Security aspects in publish/subscribe systems. In *Workshop on Distributed Event-based Systems (DEBS'04)*, May 2004.

[14] Ludger Fiege, Felix C. Grtner, Oliver Kasten, and Andreas Zeidler. Supporting mobility in Content-Based publish/subscribe middleware. In *Proc. of Middleware 2003*, pages 103–122, June 2003.

[15] P. Kabus, W. Terpstra, M. Cilia, and A. Buchmann. Addressing Cheating in Distributed Massively Multiplayer Online Games. In *Proc. of Intl Workshop on NetGames*, October 2005.

[16] S. Kounev and A. Buchmann. Performance Modeling and Evaluation of Large-Scale J2EE Applications. In *Proc. of Intl Conf of the Computer Measurement Group (CMG) on Resource Management and Performance Evaluation of Enterprise Computing Systems*, December 2003.

[17] S. Kounev and A. Buchmann. Performance Modelling of Distributed E-Business Applications using Queuing Petri Nets. In *Proc. of the IEEE Intl Symp on Performance Analysis of Systems and Software (ISPASS'03)*, March 2003.

[18] S. Kounev and A. Buchmann. SimQPN - a tool and methodology for analyzing queueing Petri net models by means of simulation. *Performance Evaluation Journal*, 2006. (To appear).

[19] C. Liebig, M. Cilia, and A.P. Buchmann. Event Composition in Time-dependent Distributed Systems. In *Proc. of CoopIS'99*, 1999.

[20] C. Liebig, M. Malva, and A.P. Buchmann. Integrating Notifications and Transactions: Concepts and X^{2}TS Prototype. In *Proc. of EDO*, LNCS 1999, 2000.

[21] Christoph Liebig and Stefan Tai. Middleware mediated transactions. In *Proc. of DOA'00*, 2001.

[22] G. Mühl, L. Fiege, and A.P. Buchmann. Filter Similarities in Content-Based Publish/Subscribe Systems. In *Proc. of ARCS*, LNCS 2299, 2002.

[23] G. Mühl, L. Fiege, F.C. Gärtner, and A.P. Buchmann. Evaluating advanced routing algorithms for content-based publish/subscribe systems. In *Proc. IEEE/ACM MASCOTS'02*, 2002.

[24] Object Management Group (OMG). Transaction service v1.1. Technical Report OMG Document formal/2000-06-28, OMG, May 2000.

[25] SPEC. The SPECjAppServer2004 Project, 2004. `www.spec.org/jAppServer2004`.

[26] J. Steffan, L. Fiege, M. Cilia, and A. Buchmann. Scoping in wireless sensor networks. In *Intl Workshop on Middleware for Pervasive and Ad-Hoc Computing (MPAC'04)*, October 2004.

[27] J. Steffan, L. Fiege, M. Cilia, and A. Buchmann. Towards Multi-Purpose wireless sensor networks. In *Proc. of Conf. on Sensor Networks (SENET'05)*, August 2005.

[28] W. Terpstra, S. Behnel, L. Fiege, J. Kangasharju, and A. Buchmann. Bit Zipper Rendezvous – Optimal Data Placement for General P2P Queries. In *EDBT 04 Workshop on Peer-to-Peer Computing & DataBases*, March 2004.

[29] A. Zeidler. *A Distributed Publish/Subscribe Notification Service for Pervasive Environments*. Ph.D. Thesis, Department of Computer Science, Darmstadt University of Technology, Germany, 2004.