INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team Sardes

System Architecture for Reflective Distributed Computing Environments

Rhône-Alpes

THEME COM

Activity Report

2005

Table of contents

1. Team

SARDES *is a project of Inria Rhône-Alpes and a research team of Imag-LSR (Software, Systems and Networks Laboratory), a joint research unit (UMR 5526) of Centre National de la Recherche Scientifique, Institut National Polytechnique de Grenoble and université Joseph Fourier.*

Head of Project-team
Jean-Bernard Stefani [Inria Research Director, on leave from *corps des Télécommunications*]

Administrative Assistant
Élodie Toihein

Inria Personnel
Daniel Hagimont [Research Scientist, till Aug. 31, 2005]
Alan Schmitt [Research Scientist]

Faculty
Sara Bouchenak [associate professor, université Joseph Fourier]
Fabienne Boyer [associate professor, université Joseph Fourier]
Noël De Palma [associate professor, Institut National Polytechnique de Grenoble]
Sébastien Jean [associate professor, université Pierre Mendès France]
Sacha Krakowiak [professor, université Joseph Fourier]
Jacques Mossière [professor, Institut National Polytechnique de Grenoble]

Post-doctoral Fellow
Adrian Mos [University College, Dublin, Ireland, from Jan. 15, 2005]

Technical Staff
Matthieu Leclercq [till Dec. 31, 2005]
Julien Legrand [from Oct. 1, 2005]
Nicolas Modrzyk [till May 15, 2005]

Ph.D. Students
Takoua Abdellatif [industrial grant (Bull)]
Philippe Bidinger [Inria grant (contract), till Nov 15, 2005]
Jakub Kornaś [Inria grant (contract)]
Renaud Lachaize [Teaching and Research Assistant (ATER), université Pierre Mendès France, until Aug. 30, 2005]
Oussama Layaida [Teaching and Research Assistant (ATER), Institut National Polytechnique de Grenoble]
Ali Erdem Özcan [industrial grant (STMicroelectronics)]
Juraj Polakovic [France Telecom R&D]
Jeremy Philippe [Inria grant (contract), from Oct. 1, 2005]
Vivien Quéma [government grant, till Dec. 31, 2005]
Sylvain Sicard [Inria grant (contract), from Oct. 1, 2005]
Christophe Taton [government grant, from Oct. 1, 2005]

Interns
Ada Diaconescu [Ph.D. student at University College, Dublin, Ireland, SARDES intern from Feb. 1, 2005 till Dec 31, 2005]
Michaël Lienhardt [4th year student at École Normale Supérieure de Cachan, from Oct. 1, 2005]

2. Overall Objectives

2.1. Goals and Organization

The overall goal of the SARDES project is to develop concepts, methods and tools to build distributed systems (primarily, software infrastructure - i.e. operating systems and middleware) that are open, evolvable and reliable. The designers and developers of such systems are facing considerable challenges, among which dynamicity, scalability, system management complexity, and quality of service. To respond to these challenges, distributed systems need to be highly adaptable and self-manageable. The project relies on the development and use of reflective and component-based software engineering techniques to achieve adaptability and to build self-manageable systems.

The project has two main ambitions:

1. To develop rigorous programming models and supporting software engineering techniques to build efficient, adaptable distributed systems.
2. To apply the concepts, models and techniques developed in the project to the prototyping of software infrastructures for self-manageable systems and to the development of automated distributed system management functions.

In line with these ambitions, the project is organized around two main research themes: *Reflective component technology*, and *Autonomous distributed systems management*.

2.2. Collaborations and Personnel Flow

SARDES is involved in several industrial and international collaborations. It is active in the OBJECTWEB consortium (8.2) dedicated to open source middleware. It is a partner of several European projects and networks: Mikado, CoreGrid, Gorda (IST program); Osmose and S4All (ITEA Eureka program). It actively participates in the national research network on software technologies (RNTL), with one ongoing project (Inside), and two new projets (SelfWare and JOnES) selected for funding in the 2005 call for proposals. It collaborates with several industrial partners: Bull, France Telecom, Microsoft, STMicroElectronics, and has close links with Scalagent, a technology startup created by former members of the project.

Note that two researchers have left the project team in 2005, to exploit in different contexts the competence they have acquired in the project.

- Emmanuel Cecchet was recruited by Emic Networks (an American company that provides cluster-based solutions for open source middleware), to develop an R&D center in Grenoble.
- Daniel Hagimont took a position of Professor at Institut National Polytechnique de Toulouse.

3. Scientific Foundations

3.1. Introduction

In this section, we first present the main challenges that face the designers of large scale distributed systems (3.2). We then discuss recent advances and open problems in the two main areas covered by SARDES: component-based architectures for adaptable distributed systems (3.3), and distributed system management (3.4).

3.2. Challenges of Distributed Systems

The future[1] of information processing applications is envisioned as a range of environments in which processors will be everywhere and will be interconnected by an array of networks, from ad-hoc to the global Internet. Constructing the software base - the *middleware* - for such ubiquitous computing infrastructures poses a number of scientific and technical challenges, which arise from the wide variety of applications and services that need to be supported.

[1]The text of 3.2 has been adapted from a Research Roadmap Report produced by the Midas IST project in which SARDES has participated.

Software will run in a multitude of computing environments ranging from traditional desktops to multiple host, networked systems, to mobile systems, to embedded systems, to wearable computers. Software systems will need to be "aware" of their physical surroundings, taking input from real-world sensors and sending output intended to control the real-world environment. They will need to "understand" their proximity to other computing resources and use this information to guide their behavior. A number of application scenarios illustrating these new environments have been discussed in a recent report for the IST European program [53].

A fundamental requirement for such environments is for middleware architectures capable of supporting services that are *composable* and *adaptable*. We have singled out four major challenges: scalability, quality of service, manageability and programmability.

1. *Scalability* concerns the ability to scale in several dimensions; scaling in machine forms: from smart labels to server farms to meta-computing network overlays; scaling in numbers: objects, machines, users, locations; scaling in logical and organizational structures: from ad-hoc collaborations networks to federations of multi-domain enterprises.
2. *Quality of service (QoS)* concerns the ability to obtain correctness and service-level guarantees such as timeliness, availability, fault-tolerance, survivability for applications executing in large scale environments. The problem of meeting QoS requirements of applications is made harder in a ubiquitous computing environment where new services and customized services are expected to be added into (existing) applications at an alarming rate.
3. *Manageability* concerns the ability to monitor and to control the operation and evolution of large scale, long-lived distributed applications and services, avoiding manual intervention and centralization (i.e., assumption of one management domain). Distributed applications and services will need to be reconfigured dynamically, for example, to maintain user specified QoS guarantees despite changes in operating conditions (e.g., component failures). Mechanisms are needed to dynamically add, extend, remove or move component services in a dependable and predictable manner.
4. *Programmability* concerns the ability to compose new applications from existing applications and services, to deploy and to maintain them in highly dynamic and heterogeneous computing environments; further, applications are expected to be highly parallel, requiring high-level abstractions necessary for dealing with complexity, with fine-grained resource awareness built according to the end-to-end principle.

Middleware for a ubiquitous computing environment will by necessity be open and standardized, at least at the level of communication protocols, key application programming interfaces and language tools. Furthermore, such middleware will become an economic commodity, i.e. be available at very low or no cost in all manners of equipment and networks to serve as a viable basis for value-added services and applications. In this context, it makes both technical and economical sense to pursue the development of such middleware facilities using an open source model.

In the next two sections, we discuss specific aspects relevant to the current research interests of SARDES.

3.3. Architectures for Adaptable Distributed Systems

A first requirement for adaptable middleware is a modular architecture, in which the interfaces and dependencies between the parts should be clearly identified. The main motivations are to minimize the adaptation induced changes by defining components as adaptation localities, and to use structural modifications (e.g. reconfiguration) as a tool for adaptation. Thus research on adaptable middleware is essentially done in the context of component-based systems. This is the theme of Section 3.3.1.

Our approach to the design and implementation of adaptable, component-based systems is presented in Section 3.3.2

3.3.1. Adaptable Component-based Middleware

Building an application out of composable parts is an old challenge of software engineering. Despite the apparent simplicity of its requirements, progress towards this goal has been slow. The notions related to components have only been elicited recently [72]. Although commercial infrastructures based on components are now available (e.g. COM+, CCM, EJB, .Net), the rigorous definition of a component model and of its architectural invariants is still an open issue.

Three main approaches are being used to achieve adaptability and separation of concerns in middleware systems: meta-object protocols, aspect-oriented programming, and pragmatic techniques.

Meta-object protocols [67] are the basic device of reflection. A reflective system is one that is able to answer questions about itself and to modify its own behavior, by providing a causally connected representation of itself to a meta-level. In a component infrastructure, the interface available at the meta-level allows the operations related to component lifecycle and composition to be dynamically adapted. A number of research prototypes (OpenORB [57], Flexinet, 2K/DynamicTAO) have been developed to investigate this approach.

The goal of aspect-oriented programming [68] (AOP) is to define methods and tools to better identify and isolate the code related to the various "aspects" present in an application, such as those related to extra-functional properties (security, fault tolerance, performance). This code is usually intertwined with that of the application proper, which is detrimental to easy change.

Pragmatic techniques usually rely on interception (interposing code at the interface between components, while preserving the interface). They are mainly used in commercial middleware.

The price of the increased flexibility brought by these adaptation techniques is paid in performance, due to the additional indirection or interpretation levels that these techniques introduce. Thus optimization techniques for reducing this performance penalty are an active subject of research. They include partial evaluation and program specialization [69], code injection (inlining), dynamic code generation and optimization [71].

3.3.2. Our Approach to Component-based Systems

The main goal in the Reflective Component Technology theme in SARDES is to develop a provably sound and efficient reflective software component technology for the construction of highly reconfigurable distributed systems, and in particular distributed software infrastructures (operating systems and middleware).

The approach taken in this theme combines a component-based approach to software construction with reflective techniques to serve as a basis of a component-based distributed programming model. In the approach, components can be used as units of encapsulation (e.g. for the purposes of modular system construction and fault isolation), and units of configuration (e.g. for the purposes of deployment, dynamic update and adaptation). Reflective components, i.e. components that expose meta-object protocol interfaces to allow for instrospection and intercession, provide the basis for constructing dynamically configurable systems.

Work on this theme places a strong emphasis on developing programming models with a formal (mostly operational) semantical basis. We believe this is essential both to ensure a sound design, but also to serve as a basis for the construction of provably safe and secure systems.

Our reflective component-based approach to software systems construction combines with an overall "exo-kernel" philosophy for the design of software infrastructures. In essence, this design philosophy emphasizes the need to refrain from building software infrastructures with too many pre-determined abstractions and pre-defined services, in order to maximize flexibility and configurability in said infrastructures. The component model that embodies these principles is FRACTAL [4], jointly developed with France Telecom R&D. It allows unlimited hierarchical composition with sharing (i.e. a component may be shared between several enclosing components). A component is has a control interface that allows managing its properties (lifecycle, attributes, contents, bindings) through appropriate controllers. These controllers may themselves be adapted and extended.

This theme includes the following research activities.

1. Models and foundations.

In this work, the aim is to develop novel models for component-based distributed programming, grounded on a rigourous semantical basis. The approach is to exploit and develop techniques from process calculi (e.g. behavioral equivalences, type systems) to formally characterize the main primitives in our programming models, and to serve as a basis for establishing the correctness of supporting tools (such as abstract machines, type checkers or code generation tools).
The main scientific challenges that we address, in this area, include: support for behavioral reflection, bisimulation-based theory for higher-order calculi, dependent type systems for higher-order calculi with localities, support for handling failures and recoverable activities, provably correct abstract machines for high-order component calculi, resource-aware components, and multi-stage programming.

2. Support for component-based distributed programming.
   In this work, the aim is to prototype basic support for component-based distributed programming. Ongoing investigations include the continued development of, and extensions to, the FRACTAL reflective component model, the definition of dynamic Architecture Description Languages (dynamic ADLs, i.e. ADLs that support the specification of component behavior and the description of reactive reconfigurations), the development of retargettable code generation tools for dynamic ADLs, the use of type systems to enforce component encapsulation constraints, and the development of optimization techniques such as proxy in-lining for run-time component structures.
   The main scientific challenges that we address, in this area, include: type systems and effective type checking for dynamic component structures, retargettable code generation for dynamic ADLs, reversible techniques for on-line component optimization, support for new target hardware architectures.

3.4. Autonomous Distributed System Management

Management (or *Administration*) is the function that aims at maintaining a system's ability to provide its specified services, with a prescribed quality of service. In general terms, administration may be viewed as a *control* activity, involving an event-reaction loop: the administration system detects events that may alter the ability of the administered system to perform its function, and reacts to these events by trying to restore this ability. The operations performed under system and application administration include configuration and deployment, reconfiguration, resource management, observation and monitoring.

Up to now, administration tasks have mainly been performed by persons. A great deal of the knowledge needed for administration tasks is not formalized and is part of the administrators' know-how and experience. As the size and complexity of the systems and applications are increasing, the costs related to administration are taking up a major part of the total information processing budgets, and the difficulty of the administration tasks tends to approach the limits of the administrators' skills. The traditional approach based on the manager-agent model, the base of widely used management protocols and frameworks such as SNMP and CMIS/CMIP, is also showing its inability to cope with the current highly dynamic managed systems.

The above remarks have motivated a new approach, in which a significant part of management-related functions is performed automatically, with minimal human intervention. This is the goal of the so-called *autonomic computing* movement [65].

Autonomic computing aims at providing systems and applications with self-management capabilities, including self-configuration (automatic configuration according to a specified policy), self-optimization (continuous performance monitoring), self-healing (detecting defects and failures, and taking corrective actions), and self-protection (taking preventive measures and defending against malicious attacks).

Several research projects [51] are active in this area. [66] is a recent survey of the main research problems related to autonomic computing.

Section 3.4.1 examines a few issues in autonomous system management. Section 3.4.2 presents our approach to these problems.

3.4.1. Selected Issues in Autonomous System Management

We examine three specific aspects related to the current activity of the project: event distribution, resource management, and fault tolerance.

3.4.1.1. Event Channels

Many management functions (monitoring, resource management, fault tolerance) rely on *events*, propagated from the administered components of the system to the management components. In a large Internet-based system, these events may be generated at a very high rate. Thus efficient channels are needed to collect and to filter these events, and to propagate them to their recipients. This problem has emerged as a topic of active research. It is also related to the fast-developing area of sensor networks.

Event propagation usually follows the publish-subscribe pattern [63]. Scalable solutions, based on multicast groups, are known for publish-subscribe based on a fixed (or slowly changing) list of topics. However, designing an efficient publish-subscribe system based on the contents of the propagated messages is still an open problem. For Internet-scale systems, even filtering algorithms running in linear time with respect to the number of subscriptions are considered inefficient, and the goal is to design sub-linear algorithms. The Gryphon [55] and Siena [60] projects have made significant progress towards this goal. The application of event channels to large scale observation and management is the theme of the Astrolabe [73] project.

3.4.1.2. Architectural Aspects of Resource Management

The advent of resource-constrained applications, such as multimedia processing and real-time control, raised the need for delegating part of the resource management to the application, while leaving the operating system with the responsibility of fair overall resource sharing between its users. The rationale is that the application is in a better position to know its precise requirements and may dynamically adapt its demands to its needs, thus allowing global resource usage to be optimized, while guaranteeing a better service to each individual application.

Two main issues need to be considered in resource management: algorithmic (defining quality criteria, and optimizing the system's performance accordingly) and architectural (organizing the system to isolate resource-oriented concerns and to exhibit the relevant interfaces). Here we concentrate on this last issue.

The first challenge is to define relevant abstractions for resource management, both for the *resource principals*, i.e. the entities to which resources are allocated, and for the resources themselves. Thus, in the case of cluster computing, new abstractions such as "cluster reserves" and "virtual clusters", which group a dynamic set of cluster-wide resources, are being investigated. Another challenge is to define what part of resource management is delegated from the operating system to the application, and to specify the relevant interfaces.

Scheduler activations [56] is an example of early work in this respect in the area of multiprocessor scheduling. A more radical approach is illustrated by exo-kernels [62] in which the lower layer exports primitives allowing a kernel or application designer to define his own resource allocation policy. This is done at the expense of additional work, which may be alleviated by the use of appropriate frameworks, as done in THINK [7]. However, since exo-kernels give access to low level system entities, they are prone to security problems.

3.4.1.3. Tracking the Causes of Internet Services Failures

As the size of current systems and applications is increasing (in terms of number of users, hardware elements, software components, geographical scale), it is quasi certain that some part of a given system or application will be faulty at any time. Maintaining the system's availability in spite of this condition is a major challenge, and one of the goals of autonomic computing ("self-repair").

An analysis of the causes of failures of Internet services [70] shows that most of the service's downtime may be attributed to management errors (e.g. wrong configuration), and that software failures come second. However, few efforts have yet been devoted to remedy this situation.

The role of configuration and deployment has long been neglected. Current research [61], [64] aims at giving a rigorous foundation to these phases of the lifecycle, thus reducing the likelihood of misconfiguration, and opening the way to autonomous configuration and deployment.

Software errors are notoriously hard to locate and to repair. Recent work [58] has shown that a semi-empirical approach (delimit the smallest possible part of a system affected by the fault, and reboot only that part) can be quite effective, as demonstrated on a full-size platform [59].

3.4.2. Our Approach to Autonomous Systems Management

The main goal in the Autonomous Distributed System Management theme in SARDES is to develop component-based software infrastructure frameworks and tools for the control and administration of large scale, long-lived distributed systems, such as clustered application servers or Grid systems. Target application domains for our prototype autonomous management systems include large Web application servers such as J2EE servers, large information mediation servers such as enterprise service buses (ESBs), and Grids.

This theme includes the following research activities.

1. Infrastructure for system management.
   In this work, the aim is to develop tools and middleware technology to support the principled implementation of system management loops. We consider the following subjects.
   - *Instrumentation for component structures:* the aim is to develop a set of tools complementing our component-based software engineering tools with specific low-level logging, monitoring and resource accounting functions. This should provide us with the technology required to efficiently and dynamically implement sensors for our systems control and management loops.
   - *Asynchronous middleware:* the aim is to develop a dynamically configurable and scalable technology for the construction of large scale (number of events, number of components, geographical dispersion) asynchronous information dissemination channels. Such channels are crucial in system management for the efficient transport of event notifications. Challenges in this area include: devising scalable routing schemes for event dissemination, defining composable abstractions for the construction of multiple asynchronous middleware personalities, fault-tolerance and overload management.
   - *Dynamic system cartography:* the aim is to develop a multi-level, multi-grain monitoring services, able to construct and maintain during a system or application life-time, a causally connected view of the system or application being managed.
2. Distributed configuration management.
   In this work, the aim is to develop a comprehensive set of system/software deployment and configuration services, from low-level bootstrapping services for component loading and system initialization, to higher-level routing, scheduling and orchestration for complex distributed system on-line deployment and configuration, with multiple component bindings and dependencies.
   Challenges in this area include: automatic resources and services discovery, dealing with partial failures, dealing with multiple dependencies and multiple coexisting versions, controlling and optimizing configuration and deployment workflows, automated support for high-level configuration policies.
3. Autonomic capabilities
   In this work, the aim is to study the automation of various systems management functions, and to allow in particular automated performability (i.e. the combination of performance and availability) management under quality of service and service level agreement constraints. The approach we follow as a first step is both *architecture-based*, i.e. leveraging an explicit component-based structure at run-time, and *empirical*, i.e. relying mostly on the empirical derivation of performance and availability models through the instrumentation and run-time monitoring of specific experimental systems. In a second step, we plan to investigate the use of more sophisticated modelling tools and techniques, including e.g. the use of control theory techniques for deriving and synthesizing control

laws for distributed resource management and overload management. Capabilities we consider initially include: automated repair management and automated sizing, starting with cluster-size systems such as clustered Web application servers.
Repair management is a direct complement to standard fault tolerance and fault recovery techniques in distributed systems, that allows a managed system to be brought back into a target regime of behavior, through reconfiguration and resource allocation, after the occurrence of a (non malicious) fault. Challenges in automating repair management include: dealing with a mixture of hardware and software faults, as well as different fault models; automatic fault detection and diagnosis; support for self-healing; support for repair management in large scale distributed systems.
Automated sizing or self-sizing aims to adapt automatically the set of resources used by the provider of a service to the level of demand for this service, e.g. automatically acquiring or releasing resources for a Web application server, according to the load of client requests. Challenges in self-sizing include: load and overload characterization in a distributed system; identifying relevant control parameters for performance tuning; devising effective algorithms for distributed performance control and load balancing.

4. Application Domains

4.1. Links to Applications

SARDES develops generic tools for distributed applications, in the form of middleware, system kernels, and information servers. These tools are useful in application domains that have one or more of the following properties.

- Cooperation using shared distributed information;
- Mobility of users, information and services;
- Need for dynamic adaptation of infrastructures or applications;
- Use of high performance information servers.

Applications are important for a project like SARDES, in which experimental aspects play a significant part. They provide testbeds to evaluate prospective designs, and they help us establish links with industrial partners, allowing us to transfer results and to identify relevant research problems.

4.2. Current Application Areas

Keywords: *electronic commerce*, *embedded systems*, *multimedia*, *power supply*, *systems administration*, *telecommunications*.

In recent years, SARDES has been active in the following application areas:

1. telecommunications : administration of large scale networks, servers and caches for the Web, management of configurable added value services;
2. power supply : administration and monitoring of power supply networked equipment, e.g. uninterrupted power supply units.
3. embedded computing: development of custom made kernels for specific applications (robotics, real time), dynamically reconfigurable kernels;
4. multimedia applications: dynamic adaptation of a videoconferencing system for use by mobile clients;
5. electronic commerce: flexible access to remote services by mobile users, efficient transaction management.

5. Software

5.1. Introduction

Keywords: *J2EE, Java thread, benchmark, cluster, clustered databases, communication middleware, mobility, publish-suscribe, transactions.*

Software development is an important aspect of the activity of SARDES. This software serves as a testbed to apply, validate and evaluate the methods and tools developed in the project.

SARDES contributes to the development of the OBJECTWEB open source code base (see 8.2), which is accessible at http://www.objectweb.org.

5.2. JavaThread : Support for mobility and persistence of Java threads

Contact : Sara Bouchenak

We have extended the Java Virtual Machine (JVM) with a non-intrusive mechanism for capturing and restoring the complete state of a Java thread. The state may be captured at any execution stage. The capture mechanism does not impose any run time performance penalty. It relies on dynamic type inference techniques for analyzing data on the execution stack, and on dynamic de-optimization of the JIT-compiled Java code. The mechanism has been implemented in the Sun Microsystems JVM.

The *JavaThread* software is distributed in binary form under an Inria license.

See http://sardes.inrialpes.fr/research/JavaThread/

5.3. C-Jdbc: Clustered JDBC-accessed Database

Contact : see OBJECTWEB page.

JDBC™(short for Java DataBase Connectivity), is a widely used Java API for accessing virtually any kind of tabular data. C-JDBC (Clustered Java™DataBase Connectivity) is an open source database cluster middleware that allows any application to transparently access a cluster of databases through JDBC. The database is distributed and possibly replicated among several nodes and C-JDBC load balances the queries between these nodes.

C-JDBC is an OBJECTWEB project distributed under an LGPL license. C-JDBC has achieved a wide visibility, and has won the Derby Code Contest at ApacheCon US in November 2004.

See http://c-jdbc.objectweb.org

5.4. Dream, a Framework for Building Asynchronous Middleware

Contact : see OBJECTWEB page.

DREAM (see 6.3.2.1) is a component-based framework dedicated to the construction of communication middleware. It provides a component library and a set of tools to build, configure and deploy middleware implementing various communication paradigms: group communications, message passing, event-reaction, publish-subscribe, etc. DREAM builds upon the FRACTAL component framework, which provides support for hierarchical and dynamic composition.

DREAM is an OBJECTWEB project distributed under an LGPL license.

See http://dream.objectweb.org

5.5. Benchmarks for J2EE Systems

Contact : see OBJECTWEB page.

RUBIS (Rice University Bidding System) is an auction site prototype modeled after `eBay.com`. RUBIS is currently used to evaluate design patterns, application servers and communication layers scalability. RUBIS has been implemented for PHP, Servlets and Enterprise Java Beans (EJB) in 7 different versions. Ongoing activities concern JDO (Java Data Objects) and Microsoft .Net versions of RUBIS. RUBBOS is a bulletin board benchmark modeled after an online news forum. Like RUBIS RUBBOS was designed to evaluate design patterns, application servers and communication layers scalability.

Both RUBIS and RUBBOS are part of the OBJECTWEB JMOB (Java Middleware Open Benchmarking) project and are distributed under an LGPL license.

See http://jmob.objectweb.org

5.6. Jotm: Java Open Transactional Manager

Contact : see OBJECTWEB page.

JOTM is an open source transaction manager implemented in Java. It was originally part of the JONAS Enterprise Java Beans platform. Since 2002, it has been extracted from the JONAS environment and developed as an autonomous project to provide support for any distributed platform that needs transactional facilities.

JOTM implements the Java Transaction API (JTA) specification with full support on both RMI/JRMP and RMI/IIOP. It also implements the Business Transaction Protocol (BTP) to support transactions for Web Services.

JOTM is an OBJECTWEB project distributed under an LGPL license.

See http://jotm.objectweb.org

6. New Results

6.1. Introduction

In this section, we present new results in the two main areas covered by SARDES: Reflective Component Technology (6.2) and Autonomous Distributed System Management (6.3).

6.2. Reflective Component Technology

The FRACTAL component model is the base of our work on reflective component technology. Section 6.2.1 presents new results on the Kell calculus, the mathematical base for FRACTAL. The following sections present two frameworks based on FRACTAL: at the application level (6.2.2) and at the operating system level (6.2.3).

6.2.1. Models for Distributed Computing

Participants: Jean-Bernard Stefani, Alan Schmitt, Philippe Bidinger.

The M-calculus [10] has been introduced to overcome a number of deficiencies in the Distributed Join calculus, including the lack of control over migrating locations, the absence of dynamic binding, and the inability to support distributed locations (or components) with different semantics (e.g. with respect to failures or access control policies). The Kell calculus [41] has in turn been introduced to alleviate some of the perceived complexity of the M-calculus (e.g. presence of multiple routing rules for communication) and to provide a more tractable setting for the development of a bisimulation theory. The type systems introduced for these calculi solve a non-trivial problem in higher-order process calculi with process passivation, namely that of ensuring the unicity of location names (a much harder problem than ensuring name linearity in the π-calculus). An abstract machine for the Kell calculus has been implemented and proved correct. This work [22], which has received the Best Paper Award at the IFIP FMOODS 2005 conference, demonstrates that the Kell calculus can be efficiently implemented. The Kell calculus can be seen as providing a formal operational basis for the FRACTAL component model. However, the original calculus does not allow for a direct definition of component structures with sharing as is the case with FRACTAL. An extension of the Kell calculus that overcomes this limitation has been recently developed [33].

The abstract machine for the Kell calculus was the subject of the Ph.D. thesis of Ph. Bidinger [11].

6.2.2. A Framework For Building Self-adaptive Multimedia Applications

Participants: Daniel Hagimont, Oussama Layaida.

The goal of this work is to develop methods and tools for designing adaptive (or, better, self-adaptive) multimedia streaming applications. To that end, such applications must be enhanced with new features allowing them to observe themselves and their environment, to detect significant changes, and to reconfigure their behavior accordingly.

To meet this goal, we have designed PLASMA [36], a multimedia proxy framework. The proxy-based adaptation approach aims at deploying adaptation processes on dedicated intermediate nodes in a wide-area multimedia stream to transcode multimedia stream and perform content-based adaptation in real-time. This approach is well-suited to reducing resource consumption such as CPU, memory and network bandwidth without modification of end-user applications, but the adaptation on the proxy has to be dynamically configured according to the constraints of the execution environment [38]. The multimedia proxy framework relies on the FRACTAL component model for building complex adaptation processes, and on a configuration language called APSL (Adaptation Proxy Specification Language) for the specification of such architectures.

This work was the subject of the Ph.D. thesis of O. Layaida [13].

Several application scenarios have been implemented, among which a video streaming application with mobile equipments. The experimental evaluations in this context [35], [37] have demonstrated that reconfiguration can be performed with a very low overhead, while significantly improving the Quality of Service of multimedia applications as well as resource usage on mobile equipments. This work has been done in association with Microsoft Research (7.3).

6.2.3. A Framework for Building Highly Configurable Operating System Kernels

Participants: Jean-Bernard Stefani, Ali Erdem Özcan, Juraj Polakovic.

THINK [7] is a component-based programming framework for building modular and customizable operating systems and applications. THINK can be considered as an implementation of FRACTAL component model that enables component-based programming in the C language.

The projects based on THINK follow three roadmaps.

1. THINK**-on-SoC.** Multi-Processor Systems-on-Chips (MPSoCs), integrating multiple processors and hardware accelerators in a single chip, seem to be the successors of classical Application-Specific Integrated Circuits (ASIC) to meet programmability, performance and energy consumption requirements of mobile devices. Implementing complex multimedia applications and networking services on such highly-constrained embedded devices is a challenging task. THINK-on-SoC addresses this issue by implementing a light-weight component-based OS and application development framework, an extension of the existing THINK framework. This environment includes transparent remote component invocations and specific component abstractions for modelling the mapping of an OS or an application on a multi-processor platform. THINK-on-SoC also explores some compile-time optimizations to improve memory and runtime performance by shrinking customized applications to fit the budget determined by the underlying MPSoC. For experimentations, we use a realistic MPSoC simulation platform provided by STMicroelectronics. The main results obtained in 2005 are described in [38] and [43].

2. Dynamic Reconfiguration. This project aims at providing dynamic reconfiguration features in operating systems and applications developed in the THINK framework. Dynamic reconfiguration involves the capture of the quiescent state of the target sub-system, the addition, removal and binding of relevant components and the state transfer from the old sub-system to the new one. We propose an original component-based solution [40]: (i) using an ADL for describing the behavior of components as a set of reactions; and (ii) providing an atomic event-based execution model for the execution of components. In 2005, we have done preliminary work on two aspects: behavioral description extensions to THINK's ADL; and implementation of two existing alternative solutions, namely MMLite and K-42, to provide a base for comparison and evaluation.

3. Multitarget FRACTAL. FRACTAL is a language independent component model, with current implementation in Java, C and C++. The goal of the Multitarget FRACTAL project is to regroup different implementations of FRACTAL within a single development environment. More precisely, our goal is to develop a unique ADL compiler that takes as input a unified ADL description and component implementations written in one of the above languages, and builds executable components for a specified platform (JVM for Java or a given hardware platform for C/C++). The compiler developed in the last quarter of 2005 takes in account two implementations of FRACTAL: Julia and THINK. Short-term plans include new implementations, as well as interoperability issues between components written in different languages.

The work on configurable component-based operating systems kernels is the subject of the Ph.D. theses of A. E. Özcan and J. Polakovic.

6.3. Autonomous Distributed System Management

This section presents new results on distributed systems management. Sections 6.3.1 and 6.3.2.1 deal with tools for management (observation and event distribution, respectively). The two following sections describe management frameworks for autonomous cluster-based Internet services (6.3.3), and for reconfigurable storage systems (6.3.4).

6.3.1. Distributed Observation and Monitoring

Participants: Renaud Lachaize, Oussama Layaida, Vivien Quéma.

A resource monitoring application needs to have the following properties: distribution, as resources are spread all over the world; scalability, as thousands of devices are involved; flexibility, as new devices are joining and leaving the system dynamically; configurability, as the monitoring application must support a wide spectrum of devices; adaptability, as operating conditions may vary; interoperability, as it is necessary to export monitored data to third-party applications.

We[2] have designed a monitoring framework that aims at meeting these challenges. It involves three kinds of distributed cooperating tasks: *Collection* tasks are deployed on monitored devices, and produce events. *Processing* tasks are in charge of processing, i.e. filtering, forwarding or aggregating events sent by collection tasks. Finally, *Presentation* tasks make events sent by processing tasks available to the observers. These tasks are built as components executing asynchronously and communicating using a message-oriented middleware (MOM) built with the framework described in 6.3.2.1.

This design is embodied in LEWYS [29], a monitoring infrastructure providing a set of hardware and software probes, monitoring filters and a repository to record monitoring information. Distributed observers can subscribe to monitoring events, resulting in the dynamic deployment of probes and communication channels. The monitoring repository is a specific observer that records monitoring events for further asynchronous processing. An experience with J2EE is described in [15].

LEWYS is available at http://lewys.objectweb.org

6.3.2. Component-based Frameworks for Dynamically Adaptable Middleware

Participants: Jean-Bernard Stefani, Takoua Abdellatif, Philippe Bidinger, Jakub Kornaś, Matthieu Leclercq, Vivien Quéma, Alan Schmitt.

Currrent Internet-based services require permanent availability and continuous evolution. These two demands may be conflicting if a server needs to be stopped for upgrade. Thus the issue of automated dynamic reconfiguration takes an increasing importance.

Two efforts within Sardes deal with the problems of dynamic reconfiguration, both using a component approach based on FRACTAL.

6.3.2.1. A Framework for Building Asynchronous Middleware

DREAM [39], [17] is a component-based software framework, built with FRACTAL, which allows the construction of different forms of asynchronous middleware, especially those based on the *publish-subscribe* pattern. DREAM has been used to re-engineer JORAM, the OBJECTWEB publish-subscribe system, bringing full configurability to JORAM at negligible expense in performance. Compared to other communication frameworks such as Click, Coyote, Ensemble, Appia, or server resource frameworks such as SEDA, DREAM provides dynamic configurability (generally not available with the above frameworks) and a smooth integration of resource management policies by means of the reflective features of the supporting FRACTAL framework.

[2]The design has been initiated in 2004 by E. Cecchet, a former member of the SARDES project team.

The DREAM framework has recently been used to develop a novel group communication protocol that provides a uniform atomic multicast service in cluster environments (joint work with École Polytechnique Fédérale de Lausanne).

The design and implementation of DREAM were the subject of the Ph.D. thesis of V. Quéma [14].

The implementation of dynamic reconfiguration in DREAM was a motivation for adding dynamic aspects to the FRACTAL ADL, a work that is still in progress. Another ongoing task is the formal specification of a type system for DREAM, which would allow verification and proofs. Preliminary results are published in [21].

DREAM is available at http://dream.objectweb.org (see also 5.4).

6.3.2.2. Packaging and Reconfiguration of Component-based Middleware

Updating a service at runtime involves three main tasks: (i) isolation of services as independent packages; (ii) deployment and redeployment of services; and (iii) handling service dependencies and state at runtime. We have investigated these three aspects, using J2EE servers as a testbed.

Starting with point (iii), we have reimplemented JONAS, an open source J2EE platform, by decomposing it in a hierarchy of parts, each of which is wrapped into a FRACTAL component. Each component thus becomes an independent unit of configuration and deployment. The new system, called JONASALACARTE, then inherits all properties of a FRACTAL-based implementation, among which dynamic configurability. Experiments [18] show that this is achieved with low overhead.

Points (i) and (ii) are also tackled using FRACTAL as a base for implementing service packaging. Dependencies are expressed using fractal bindings. The deployment infrastructure is also implemented using FRACTAL components, which allows it to be dynamically adaptable (e.g. by plugging various deployment policy modules). This work is reported in [19].

The goal of the ongoing research is to build a unified model and framework for the management tasks related to the packaging, deployment and reconfiguration process. We also need to consider such issues as versioning, as well as licensing issues (e.g. identifying independent units for the licensing aspect).

Methods and tools for component packaging and dynamic reconfiguration are the subject of the Ph.D. theses of T. Abdellatif and J. Kornaś.

6.3.3. A Component-based Framework for Autonomic Computing

Participants: Jean-Bernard Stefani, Takoua Abdellatif, Sara Bouchenak, Fabienne Boyer, Noël De Palma, Daniel Hagimont, Jakub Kornaś, Julien Legrand, Adrian Mos, Jeremy Philippe, Sylvain Sicard, Christophe Taton, Sacha Krakowiak.

In 2004, we have started the design of a platform called JADE, which we plan to use as a testbed for research on autonomous systems management, integrating most of the work described in the previous sections. The infrastructure of JADE is a LAN-based cluster of Unix nodes. The application domain is interactive, data-driven computing based on dynamic web content, an area that is representative of an important segment of distributed computing, for which availability, performance, and ease of run-time administration are crucial issues. We have selected J2EE as the initial middleware environment, since we have acquired experience in the design, use and evaluation of various components of this environment, specially in the context of OBJECTWEB (8.2).

The main thesis that we defend is that *architecture-based management is an adequate base for autonomic computing*. We therefore need (i) an explicit architectural description of both the managed and the managing elements; and (ii) explicit management interfaces for the various elements at all levels of description.

To that end, the current version of JADE uses the FRACTAL model, which answers both above requirements: the overall structure of the system is described by means of an ADL; and each component has explicit management interfaces allowing, among others, dynamic rebinding and reconfiguration. In order to integrate legacy systems (such as the tiers that make up a J2EE platform) in a FRACTAL environment, we use wrapping techniques that embed the management operations of the legacy system into FRACTAL controllers. As an important side effect, the task of deploying such a “fractalized” application is greatly simplified (the size of the needed command scripts is reduced by an order of magnitude).

Up to now, we have investigated two aspects of autonomic management using the JADE platform.

- *Automated recovery management in the face of failures* (physical node failures in a first step; software failures in a later phase). The recovery manager relies on an explicit, causally connected description of the running system; these services are themselves supervised in the same way as the managed J2EE system. To that end, we exploit our work on architectural description of component-based systems. The internal representation is used by the control loops that detect and analyze abnormal conditions, and attempt to correct them by means of actuators performing recovery tasks. Providing spare resources are available, the first experiments show that the system adequately reacts to a node failure and recovers its initial performance. Results are reported in [49], [24], [26], [23].
- *Automated resource management in the face of unexpected variations of the load*. In a first phase, we have essentially investigated node allocation in a cluster. Preliminary results are reported in [50], [42]. We plan to extend this work to other resources, and to examine the use of QoS related indices to define targets and evaluation criteria for autonomous resource management policies [48].

Recalling the intended role of the JADE platform as a testbed for autonomous systems management, we plan to integrate the results of our research on measurement (6.3.1) and asynchronous communication (6.3.2.1). The work on packaging and configuration of component-based systems is also relevant (see [47], [46]).

Three Ph.D. theses have been initiated in 2005 on various aspects of autonomous management: J. Philippe (QoS), S. Sicard (recovery), and Ch. Taton (resource management).

Current work includes: improving and documenting the overall design of the JADE platform; investigating autonomic management of security (reacting to intrusion attempts); applying aspect-oriented techniques to autonomic management [25]. A longer term perspective is the use of methods and tools based on control theory, in collaboration with control specialists.

6.3.4. A Framework for Building Reconfigurable Storage Systems

Participants: Jacques Mossière, Renaud Lachaize.

The aim of the Proboscis[3] project is to study the sharing of storage devices in clusters, where the storage devices are distributed across the nodes and accessed with efficient networking technologies. For this purpose, we are developing a modular and extensible software infrastructure for remote disk access construction and administration. This framework is intended as a set of building blocks, on which specific storage services such as cluster file systems can be built and deployed. A key feature of Proboscis lies in its use of explicit I/O data paths to facilitate disk access scheduling, reconfiguration at runtime and monitoring.

Recent work on Proboscis has focused on dynamic reconfiguration of the data paths to increase the flexibility and availability of clustered data servers. The modularity of the framework is used to allow non-disruptive changes of the low level storage services without any impact on the upper layers of the system (e.g. logical volume managers, cluster file systems or databases). This is beneficial for fault tolerance (data redistribution in case of a server crash), for performance (load balancing and caching policies), and for occasional administration needs (e.g. code updates or transparent migration to new storage or networking hardware). We have leveraged some of the adaptable and autonomic features of the framework for realistic use cases such as distributed swapping and cooperative caching.

Beyond specific support for storage systems, we also have developed more general techniques for component-based applications deployed on clusters. In particular, we have proposed a programming model to abstract the way in which data buffers are exchanged between nodes. This allows communications to be optimized according to various static and dynamic parameters (e.g. available hardware and protocols, I/O load) without any modification to the code of the application [34].

This work was the subject of the Ph.D. thesis of R. Lachaize [12].

The source code of Proboscis is available on request. See http://sardes.imag.fr/~rlachaiz/ proboscis.shtml.

[3]Proboscis is a joint research effort of SARDES and the DistLab group at DIKU, the Computer Science Laboratory of the University of Copenhagen.

7. Contracts and Grants with Industry

7.1. Collaboration with Bull

Participants: Jacques Mossière, Takoua Abdellatif.

The theme of the collaboration with Bull is the development of system software for exploiting clusters operating under Linux for scientific and data management applications. This collaboration started in October 2000, and also involves the Inria Mescal project-team (Yves Denneulin).

Our contribution is the development of an administration support system for a cluster, with a specific emphasis on J2EE platforms. This work started in September 2003 and is the subject of the Ph.D. thesis of T. Abdellatif (Bull co-funding).

In 2005, we have experimented with a "fractalized" version of JONAS, an open source J2EE platform developed at Bull and distributed by the OBJECTWEB consortium. The resulting system (called JONASALACARTE) thus becomes easily extendable and adaptable, with low overhead. See 6.3.2.2 for more details and references.

7.2. Collaboration with France-Telecom R&D

Participants: Jean-Bernard Stefani, Vivien Quéma, Jakub Kornaś.

SARDES maintains an active collaboration with France-Telecom R&D (Norbert Segard Center, Distributed Systems Architecture group):

1. Collaboration on the following aspects: extensions to the FRACTAL component model [4]; framework for dynamic reconfiguration of FRACTAL based applications (J.-B. Stefani, J. Kornaś, V. Quéma). Part of this work takes place within the OBJECTWEB consortium (8.2); see also 6.3.2.2.
2. Collaboration on distributed process calculi including mobility and distribution (6.2.1), and on the associated virtual machines (J.-B. Stefani). This work is done within the European IST project Mikado (8.3.1).

7.3. Collaboration with Microsoft

Participants: Daniel Hagimont, Oussama Layaida.

The goal of this contract is to develop techniques for dynamic adaptation of embedded multimedia applications on Microsoft Windows CE .Net. This work is jointly carried out by project-teams SARDES and WAM at Inria Rhône-Alpes (N. Layaida).

We have extended the adaptation techniques developed in SARDES for dynamic proxy configuration in order to ensure QoS of multimedia applications using the Microsoft DirectShow middleware framework. We have shown that our techniques can be applied in a resource-constrained environment. See also 6.2.2.

7.4. Collaboration with STMicroElectronics

Participants: Jacques Mossière, Jean-Bernard Stefani, Ali Erdem Özcan.

The goal of this project, started in November 2003, is to investigate the use of the THINK framework to develop operating systems for on-chip multiprocessors. See 6.2.3 for a more detailed description.

This work is the subject of the Ph.D. thesis of A. E. Özcan (STMicroElectronics co-funding).

7.5. RNTL Inside Project

Participants: Jean-Bernard Stefani, Matthieu Leclercq, Vivien Quéma.

Inside is a pre-competitive project funded by the French Ministry of Industry under the RNTL program. Its partners are Inria (SARDES project-team), and three companies: Schneider Electric, OpenSugar and ScalAgent. The goal of the project is to develop a software infrastructure to support Internet-based services (specially distributed equipment monitoring) in the area of power distribution. The infrastructure was validated in a real-life environment. The contribution of SARDES is the work on event-based middleware, including the DREAM prototype (see 5.4). The project ended in November 2005.

See http://www.telecom.gouv.fr/rntl/projet/Posters-PDF/RNTL-Poster-INSIDE.pdf

8. Other Grants and Activities

8.1. National Actions

8.1.1. ARP Network

SARDES is a member of the "Distributed Systems and Applications" group of the national research network on Systems Architecture, Networks and Parallelism (ARP: Architecture, Réseaux et Parallélisme).

See http://www.arp.cnrs.fr.

8.2. ObjectWeb Consortium

Participants: Jean-Bernard Stefani, Sacha Krakowiak.

OBJECTWEB is an open-source software community created at the end of 1999 by France Telecom R&D, Bull and Inria. Its goal is the development of open-source distributed middleware, in the form of flexible and adaptable components. These components range from specific software frameworks and protocols to integrated platforms. OBJECTWEB developments follow a systematic component-based approach.

In 2002, OBJECTWEB evolved into an international consortium hosted by Inria. The consortium is an independent non-profit organization open to companies, institutions and individuals. An account of the OBJECTWEB transfer model and experience may be found in [28].

SARDES contributes to OBJECTWEB through its technical involvement in the development of software components and frameworks (e.g. FRACTAL, C-JDBC, JOTM, DREAM, TRIBE) and through participation in the management structures of the consortium: board (J.-B. Stefani, past president and current member), and college of architects (J.-B. Stefani, current member and S. Krakowiak, member till March 2005). The president of the college of architects in 2005 was E. Cecchet, a former member of the SARDES project team.

See http://www.objectweb.org

8.3. Projects Funded by the European Commission

8.3.1. IST Project Mikado

Participants: Jean-Bernard Stefani, Alan Schmitt, Philippe Bidinger.

The goal of the IST project Mikado is to develop a model for distributed mobile programming based on a process calculus. Investigation topics include type models, programming language and virtual machines technologies, and specification languages. The project's partners are: Inria (project-teams Mimosa and SARDES), France Telecom R&D (DTL/ASR Laboratory), University of Sussex, UK (M. Hennessy), Università di Firenze, Italy (R. de Nicola), University of Lisbon, Portugal (V. Vasconcelos). The main contribution of SARDES is in the areas of models for component-based systems and distributed process calculi. The project ended in mid-2005.

See http://mikado.di.fc.ul.pt/

8.3.2. IST Project Gorda

Participants: Jean-Bernard Stefani, Sara Bouchenak.

GORDA (Open Replication of Databases) is an IST STREPS project. The GORDA project aims at (i) promoting the interoperability of databases and replication protocols by defining generic architecture and interfaces that can be standardized; (ii) providing general purpose and widely-applicable database systems; and (iii) providing uniform techniques and tools for managing secure and heterogeneous replicated database systems. The project partners are Universidade do Minho (R. Oliveira), Università della Svizzera Italiana (F. Pedone), Universidade de Lisboa (L. Rodrigues), INRIA (SARDES project), Emic Networks (Finland) and MySQL AB (Sweden). The main contribution of SARDES is the C-JDBC technology, and the development of frameworks and tools for its use.

See http://gorda.di.uminho.pt

8.3.3. ITEA Project Osmose

Participant: Jean-Bernard Stefani.

Osmose (Open Source Middleware for Open Systems in Europe) is a project funded by the ITEA program. The project is focused on the development, enhancement, and validation in defined test-beds of a comprehensive adaptable Open Source middleware to be hosted by the OBJECTWEB consortium (8.2). The testbeds will be developed in three areas: telecom services, home gateway, and avionics.

The Consortium is built around three sets of partners: 6 large industrial companies (Bull, France Telecom, Philips, Telefonica, Telvent and Thales), 6 SMEs (Bantry Technologies, iTEL, Kelua, Lynx, VICORE and Whitestein Technologies), and 7 academic partners (Charles University, EPFL, Inria, INT, LIFL, Imag-LSR and Universidad Politécnica de Madrid). The contribution of SARDES is in the area of frameworks and tools for component-based middleware. The project ended in October 2005.

See http://www.itea-osmose.org/

8.3.4. ITEA Project S4All

Participant: Jean-Bernard Stefani.

S4All (Services for All) is a project funded by the ITEA program. The project, launched in July 2005 for a duration of 2 years, is focused on the provision of a platform for easy creation of person-to-person communication and customized web services. The platform is organized as a “service bus”, allowing flexible integration of existing applications, as well as new developments.

The Consortium includes Alcatel, Bull and Inria. The contribution of SARDES is expertise in the area of adaptive middleware and integration of applications.

8.4. International Networks and Working Groups

8.4.1. CoreGrid Network of Excellence (IST-004265)

SARDES is a member of CoreGrid, the European Research Network on Foundations, Software Infrastructures and Applications for large scale distributed, GRID and Peer-to-Peer Technologies, launched in 2004 for four years. Its objective is to coordinate European efforts in the area of Grid and Peer-to-Peer technologies. It gathers forty-two institutions. SARDES is contributing in the areas of programming models and software architecture (FRACTAL has been adopted as a basis for the component-based programming model of CoreGrid).

See http://www.coregrid.net/

8.5. International Bilateral Collaborations

8.5.1. Europe

SARDES maintains long term collaboration with several research groups in Europe:

- École Polytechnique Fédérale de Lausanne: Distributed Systems Laboratory (Prof. André Schiper), Distributed Programming Laboratory (Prof. Rachid Guerraoui), LabOS (Prof. Willy Zwaenepoel). Collaboration on fault tolerance and performance management for clustered applications. See 6.3.2.1.
- University of Lancaster, *Distributed Media Systems* (Prof. Gordon Blair), on adaptable middleware for multimedia communication.

- Trinity College, Dublin, *Distributed Systems Group* (Prof. Vinny Cahill), on distributed programming and clusters. A. Senart, a former SARDES Ph.D. student, has been staying with TCD since academic year October 2003.
- University of Copenhagen, DIKU Laboratory *Distributed Systems Group* (Prof. Eric Jul, Dr. Jørgen Hansen), on system support for clustered servers (visits and shared software). See 6.3.4.

8.5.2. China

Sardes is actively pursuing contacts with several Chinese universites and R&D teams on middleware, both for research cooperation purposes and on behalf of the OBJECTWEB Consortium. For research cooperation, the most advanced contacts are with Peking University (Prof. Mei), on the subject of advanced architecture description languages and autonomic distributed system management, with the National University of Defense Technology (Prof. Wang), on autonomic distributed systems management, and with the South China University of Technology in Guangzhou (Prof. Xu), on J2EE systems management.

9. Dissemination

9.1. Community Service

Several members of SARDES contributed to the organization of *Middleware'05*, the main scientific conference in the area of middleware, held at Grenoble from November 28 to December 2. J.-B. Stefani is the general chair. S. Jean and N. De Palma are co-chairs of the organization committee; D. Hagimont also contributed to the organization before leaving INRIA. J. Mossière is the chair of the Doctoral Symposium associated with the Conference. Other persons helped in local organization tasks. See http://middleware05.objectweb.org/

J.-B. Stefani is a member of the editorial board of the journal *Annales des Télécommunications*, and of the program committees of the *Middleware'05* Conference, the DOA 2005 Conference (*Distributed Objects and Applications*), the *SOC-EUSAI* Conference and the TGC (TRUSTWORTHY GLOBAL COMPUTING Symposium. He also is a member of the scientific Advisory Board of NTT DoCoMO Labs, and a member of the Comité d'Évaluation of the French program RNTL (Research Network on Software Technologies).

A. Schmitt is a member of the program committee of DBPL'05 (*10th International Symposium on Data Base Programming Languages*). He is also in charge of a joint Imag-Inria seminar on "Distributed Systems and Data Management".

S. Krakowiak is a member of the steering committee of ERSADS (*European Research Seminar on Advances in Distributed Systems*), the school/workshop initiated in 1995 by the CaberNet Network of Excellence. He is also the chairman of the committee sponsored by *Specif* (the French Computer Science Researchers' and Teachers' Association) to select each year an outstanding Ph.D. thesis in Computer Science defended in France. He has done evaluation work for the University of Copenhagen, for the Danish Natural Science Research Council, and for Morgan Kaufmann Publishers.

9.2. University Teaching

S. Bouchenak, F. Boyer, N. De Palma, S. Krakowiak, and J. Mossière have taught several operating systems and distributed systems courses at the M.S. and M.Eng. levels, both at Institut National Polytechnique de Grenoble and at université Joseph Fourier. Most of our Ph.D. students contributed to these courses as teaching assistants.

9.3. Participation in Seminars, Workshops, Conferences

Several members of SARDES attended various scientific conferences and workshops. See the relevant section of the Bibliography for details.

9.4. Miscellaneous

This section mentions publications involving members of SARDES, reporting on work done in collaboration with external partners but not directly related to the project. See [20], [31], [32], [30].

Most of the publications of the project are available on line from the Sardes web site:
http://sardes.inrialpes.fr/

10. Bibliography

Major publications by the team in recent years

[1] S. KRAKOWIAK, S. SHRIVASTAVA (editors). *Recent Advances in Distributed Systems*, Lecture Notes in Computer Science 1752, Springer, 2000.

[2] G. BLAIR, J.-B. STEFANI. *Open Distributed Processing and Multimedia*, Addison-Wesley, 1997.

[3] S. BOUCHENAK, D. HAGIMONT, S. KRAKOWIAK, N. DE PALMA, F. BOYER. *Experiences Implementing Efficient Java Thread Serialization, Mobility and Persistence*, in "Software - Practice and Experience", vol. 34, n° 4, april 2004, p. 355–394.

[4] E. BRUNETON, T. COUPAYE, M. LECLERCQ, V. QUÉMA, J.-B. STEFANI. *An Open Component Model and its Support in Java*, in "Component-based Software Engineering, Proceedings of the International Symposium CBSE 2004, Edinburgh, Scotland", I. CRNKOVIC, J. A. STAFFORD, H. W. SCHMIDT, K. WALLNAU (editors). , Lecture Notes in Computer Science 3054, Springer, May 2004, p. 7–22.

[5] É. BRUNETON, M. RIVEILL. *An Architecture for Extensible Middleware Platforms*, in "Software – Practice and Experience", vol. 31, November 2001, p. 1237–1264.

[6] E. CECCHET, J. MARGUERITE, W. ZWAENEPOEL. *Performance and Scalability of EJB Applications*, in "Proceedings of OOPSLA'02, Seattle, WA, USA", November 2002.

[7] J.-PH. FASSINO, J.-B. STEFANI, J. LAWALL, G. MULLER. *THINK: A Software Framework for Component-based Operating System Kernels*, in "Proceedings of Usenix Annual Technical Conference, Monterey (USA)", June 10th-15th 2002.

[8] D. HAGIMONT, F. BOYER. *A Configurable RMI Mechanism for Sharing Distributed Java Objects*, in "IEEE Internet Computing", vol. 5, n° 1, 2001, p. 36–44.

[9] É. NAJM, A. NIMOUR, J.-B. STEFANI. *Behavioural Typing for Objects and Process Calculi*, chap. 13, Cambridge University Press, 2001, p. 281–301.

[10] A. SCHMITT, J.-B. STEFANI. *The M-calculus: A Higher-Order Distributed Process Calculus*, in "In Proceedings of the 30th Annual ACM Symposium on Principles of Programming Languages (POPL'03), New Orleans, LA, USA", January 15-17 2003.

Doctoral dissertations and Habilitation theses

[11] P. BIDINGER. *Le Kell calcul, typage et implantation*, Ph. D. Thesis, Université Joseph Fourier, Grenoble, December 2005.

[12] R. LACHAIZE. *Un canevas logiciel pour la construction de systèmes de stockage reconfigurables*, Ph. D. Thesis, Institut National Polytechnique de Grenoble, September 2005.

[13] O. LAYAÏDA. *Un support logiciel à composants pour la construction d'applications multimédia adaptatives*, Ph. D. Thesis, Institut National Polytechnique de Grenoble, December 2005.

[14] V. QUÉMA. *Vers l'exogiciel – Une approche de la construction d'infrastructures logicielles radicalement configurables*, Ph. D. Thesis, Institut National Polytechnique de Grenoble, December 2005.

Articles in refereed journals and book chapters

[15] E. CECCHET, H. ELMELEEGY, O. LAYAÏDA, V. QUÉMA. *Implementing Probes for J2EE Cluster Monitoring*, in "Studia Informatica", vol. 4, n° 1, 2005.

[16] S. KRAKOWIAK, J.-B. STEFANI. *export-bind : Un patron d'architecture pour la liaison adaptable*, in "Informatique Répartie", D. TRYSTRAM, Y. SLIMANI, M. JEMNI (editors). , Hors-série de la Revue des sciences et technologies de l'information. English version, chap. 9, Hermès, 2005, http://sardes.inrialpes.fr/papers/files/05_CFSE-export-bind.pdf.

[17] M. LECLERCQ, V. QUÉMA, J.-B. STEFANI. *DREAM: a Component Framework for the Construction of Resource-Aware, Configurable MOMs*, in "IEEE Distributed Systems Online", vol. 6, n° 9, September 2005.

Publications in Conferences and Workshops

[18] T. ABDELLATIF. *Enhancing the Management of a J2EE Application Server using a Component-Based Architecture*, in "31th IEEE/Euromicro Conference, CBSE Track: Component-based Software Engineering, Porto, Portugal", August 2005.

[19] T. ABDELLATIF, J. KORNAŚ, J.-B. STEFANI. *J2EE Packaging, Deployment and Reconfiguration Using a General Component Model*, in "3rd International Working Conference on Component Deployment (CD 2005), Grenoble, France", November 2005.

[20] C. AMZA, G. SOUNDARARAJAN, E. CECCHET. *Transparent Caching with Strong Consistency in Dynamic Content Web Sites*, in "ICS05, 19th ACM International Conference on Supercomputing, Cambridge, MA, USA", June 2005.

[21] P. BIDINGER, M. LECLERCQ, V. QUÉMA, A. SCHMITT, J.-B. STEFANI. *Dream Types - A Domain Specific Type System for Component-Based Message-Oriented Middleware*, in "4th Workshop on Specification and Verification of Component-Based Systems (SAVCBS'05), in association with ESEC/FSE'05, Lisbon, Portugal", September 2005.

[22] P. BIDINGER, A. SCHMITT, J.-B. STEFANI. *An Abstract Machine for the Kell Calculus*, in "7th IFIP

International Conference on Formal Methods for Object-Based Distributed Systems (FMOODS), Athens, Greece", Best Paper Award, June 2005.

[23] S. BOUCHENAK, F. BOYER, D. HAGIMONT, S. KRAKOWIAK, A. MOS, N. DE PALMA, V. QUÉMA, J.-B. STEFANI. *Architecture-Based Autonomous Repair Management: An Application to J2EE Clusters*, in "24th IEEE Symposium on Reliable Distributed Systems (SRDS), Orlando, Florida, USA", October 2005.

[24] S. BOUCHENAK, F. BOYER, D. HAGIMONT, S. KRAKOWIAK, N. DE PALMA, V. QUÉMA, J.-B. STEFANI. *Architecture-Based Autonomous Repair Management - Application to J2EE Clusters*, in "2nd IEEE International Conference on Autonomic Computing (ICAC-05) (Poster), Seattle, Washington, USA", June 2005.

[25] S. BOUCHENAK, N. DE PALMA, S. FONTAINE, B. TÊTE. *AOSD for Internet Service Clusters: The Case of Availability*, in "Workshop on Aspect-Oriented Middleware Development (AOMD05), at 6th International Middleware Conference, Grenoble, France", December 2005.

[26] S. BOUCHENAK, N. DE PALMA, D. HAGIMONT. *Autonomic Administration of Clustered J2EE Applications*, in "IFIP/IEEE International Workshop on Self-Managed Systems & Services (SelfMan 2005), Nice, France", October 2005.

[27] S. BOUCHENAK, N. DE PALMA, S. KRAKOWIAK. *Tolérance aux fautes dans les grappes d'applications Internet*, in "4ème Conférence Française sur les Systèmes d'Exploitation (CFSE 2005), Le Croisic, France", April 2005.

[28] E. CECCHET. *From research prototypes to industrial strength open source products - The ObjectWeb experience*, in "31st Annual Conference on Current Trends in Theory and Practice of Informatics (SOFSEM 2005), Liptovsky Jan, Slovak Republic", January 2005.

[29] E. CECCHET, O. LAYAÏDA, V. QUÉMA. *LeWYS : un Canevas Logiciel à Composants pour Construire des Applications de Supervision*, in "4ème Conférence Francophone autour des Composants Logiciels (avec CFSE-RENPAR 2005), Le Croisic, France", April 2005.

[30] J. N. FOSTER, M. B. GREENWALD, C. KIRKEGAARD, B. C. PIERCE, A. SCHMITT. *Exploiting Schemas in Data Synchronization*, in "Database Programming Languages (DBPL)", August 2005.

[31] J. N. FOSTER, M. B. GREENWALD, J. T. MOORE, B. C. PIERCE, A. SCHMITT. *Combinators for Bi-Directional Tree Transformations: A Linguistic Approach to the View Update Problem*, in "ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Long Beach, California", 2005.

[32] V. GAPEYEV, M. Y. LEVIN, B. C. PIERCE, A. SCHMITT. *XML Goes Native: Run-time Representations for Xtatic*, in "14th International Conference on Compiler Construction, Edinburgh, UK", April 2005.

[33] D. HIRSCHKOFF, T. HIRSCHOWITZ, D. POUS, A. SCHMITT, J.-B. STEFANI. *Component-Oriented Programming with Sharing: Containment is not Ownership*, in "4th International Conference on Generative Programming and Component Engineering (GPCE), Tallinn, Estonia", September 2005.

[34] R. LACHAIZE. *A Distributed Shared Buffer Space for Data-intensive Applications*, in "Proceedings of the 5th International Workshop on Distributed Shared Memory, Cardiff, UK", May 2005.

[35] O. LAYAÏDA, D. HAGIMONT. *Composition et Reconfiguration Hiérarchiques pour des Services Multimédia Auto-Adaptables*, in "Rencontres Francophones en Parallélisme, Architecture, Systèmes et Composants - CFSE, Le Croisic, France", April 2005.

[36] O. LAYAÏDA, D. HAGIMONT. *PLASMA : A Component-based Framework for Building Self-Adaptive Applications*, in "Proc. SPIE/IS&T Symposium On Electronic Imaging, Conference on Embedded Multimedia Processing and Communications, San Jose, CA, USA", January 2005.

[37] O. LAYAÏDA, D. HAGIMONT. *Designing Self-adaptive Multimedia Applications Through Hierarchical Reconfiguration*, in "The 5th IFIP Conference on Distributed Applications and Interoperable Systems, Athens, Greece", Lecture Notes in Computer Science, vol. 3543, Springer, June 2005, p. 95-107.

[38] O. LAYAÏDA, A. E. ÖZCAN, JEAN-BERNARD. STEFANI. *A Component-based Approach for MPSoC SW Design: Experience with OS Customization for H.264 Decoder*, in "3rd Workshop on Embedded Systems for Real-Time Multimedia under CODES+ISSS", 2005.

[39] M. LECLERCQ, V. QUÉMA, J.-B. STEFANI. *DREAM : un Canevas Logiciel à Composants pour la Construction d'Intergiciels Orientés Messages Dynamiquement Configurables*, in "4ème Conférence Francophone autour des Composants Logiciels (avec CFSE-RENPAR 2005), Le Croisic, France", April 2005.

[40] J. POLAKOVIC, A. E. ÖZCAN, J.-B. STEFANI. *Reconfiguration dynamique d'un système à composants*, in "4ème Conférence Francophone autour des Composants Logiciels (avec CFSE-RENPAR 2005), Le Croisic, France", April 2005.

[41] A. SCHMITT, J.-B. STEFANI. *The Kell Calculus: A Family of Higher-Order Distributed Process Calculi.*, in "Global Computing", Lecture Notes in Computer Science, vol. 3267, Springer, 2005.

[42] C. TATON, S. BOUCHENAK, F. BOYER, N. DE PALMA, D. HAGIMONT, A. MOS. *Self-Manageable Replicated Servers*, in "VLDB Workshop on Design, Implementation, and Deployment of Database Replication (WDIDDR), at 31st International Conference on Very Large Data Bases, Trondheim, Norway", August 2005.

[43] A. E. ÖZCAN, S. JEAN, J.-B. STEFANI. *Bringing Ease and Adaptability to MPSoC Software Design: a Component-Based Approach*, in "Construction and Analysis of Safe, Secure and Interoperable Smart devices (CASSIS Workshop, Nice, France", March 2005.

Internal Reports

[44] S. BOUCHENAK, A. COX, S. DROPSHO, S. MITTAL, W. ZWAENEPOEL. *AOP-Based Caching of Dynamic Web Content: Experience with J2EE Applications*, 35 pages, Technical report, n° RR-5483, INRIA, projet Sardes, February 2005, http://www.inria.fr/rrrt/rr-5483.html.

[45] S. BOUCHENAK, S. KRAKOWIAK, N. DE PALMA. *Tolérance aux fautes dans les applications J2EE : Une solution scalable et générique*, 26 pages, Technical report, n° RR-5463, INRIA, projet Sardes, January 2005, http://www.inria.fr/rrrt/rr-5463.html.

Miscellaneous

[46] S. BAUZA. *Installation automatisée d'applications sur un système distribué*, Rapport de Master Recherche Informatique : Systèmes et Logiciel (Master's thesis), École Doctorale Mathématiques et Informatique, Grenoble, June 2005.

[47] J. BOURCIER. *Déploiement d'applications J2EE sur des serveurs en grappe*, Rapport de Master Recherche Informatique : Systèmes et Logiciel (Master's thesis), École Doctorale Mathématiques et Informatique, Grenoble, June 2005.

[48] J. PHILIPPE. *Garanties de Niveau de Service pour les Applications Web*, Rapport de Master Recherche Informatique : Systèmes et Logiciel (Master's thesis), École Doctorale Mathématiques et Informatique, Grenoble, June 2005.

[49] S. SICARD. *Tolérance aux fautes pour les systèmes autonomes*, Rapport de Master Recherche Informatique : Systèmes et Logiciel (Master's thesis), École Doctorale Mathématiques et Informatique, Grenoble, June 2005.

[50] C. TATON. *Reconfiguration dynamique d'applications réparties sur une grappe de serveurs pour l'adaptation à la charge*, Rapport de Master Recherche Informatique : Systèmes et Logiciel (Master's thesis), École Doctorale Mathématiques et Informatique, Grenoble, June 2005.

Bibliography in notes

[51] *Research Projects in Autonomic Computing*, 2003, http://www.research.ibm.com/autonomic/research/projects.html, IBM Research.

[52] *Proceedings of the 18th International Conference on Distributed Computing Systems*, IEEE Computer Society Press, Amsterdam, The Netherlands, May 1998.

[53] *Report of the IST Advisory Group concerning Software Techologies, Embedded Systems, and Distributed Systems: A European Strategy towards an Ambient Intelligent environment*, June 2002, http://www.cordis.lu/ist/istag.htm, Office for Official Publications of the European Communities.

[54] *Proceedings of the 1995 ACM Symposium on Operating Systems Principles*, December 1995.

[55] M. K. AGUILERA, R. E. STROM, D. C. STURMAN, M. ASTLEY, T. D. CHANDRA. *Matching Events in a Content-Based Subscription System*, in "Symposium on Principles of Distributed Computing (PODC)", 1999, p. 53-61.

[56] T. E. ANDERSON, B. N. BERSHAD, E. D. LAZOWSKA, H. M. LEVY. *Scheduler Activations: Effective Kernel Support for the User-Level Management of Parallelism*, in "ACM Transactions on Computer Systems", vol. 10, n° 1, February 1992, p. 53–79.

[57] G. BLAIR, G. COULSON, A. ANDERSEN, L. BLAIR, M. CLARKE, F. COSTA, H. DURAN-LIMON, T. FITZPATRICK, L. JOHNSTON, R. MOREIRA, N. PARLAVANTZAS, K. SAIKOSKI. *The Design and Implementation of OpenORB v2*, in "IEEE Distributed Systems Online, vol. 2 no 6, Special Issue on Reflective Middleware", 2001.

[58] G. CANDEA, S. KAWAMOTO, Y. FUJIKI, G. FRIEDMAN, A. FOX. *Microreboot – A Technique for Cheap Recovery*, in "6th Symposium on Operating Systems Design and Implementation (OSDI), San Francisco, CA, USA", December 2004, p. 31–44.

[59] G. CANDEA, E. KICIMAN, S. ZHANG, P. KEYANI, A. FOX. *JAGR: An Autonomous Self-Recovering Application Server*, in "5th International Workshop on Active Middleware Services (AMS), Seattle, WA, USA", June 2003.

[60] A. CARZANIGA, D. S. ROSENBLUM, A. L. WOLF. *Design and evaluation of a wide-area event notification service*, in "ACM Transactions on Computer Systems (TOCS)", vol. 19, n° 3, 2001, p. 332–383.

[61] E. DOLSTRA, M. DE JONGE, E. VISSER. *Nix: A Safe and Policy-Free System for Software Deployment*, in "18th Large Installation System Administration Conference (LISA '04), Atlanta, Georgia, USA", L. DAMON (editor). , USENIX, November 2004, p. 79-92, http://www.usenix.org/events/lisa04/.

[62] D. ENGLER, M. KAASHOEK, J. O'TOOLE. *Exokernel: An Operating System Architecture for Application-Level Resource Management*, in "Proceedings of the 1995 ACM Symposium on Operating Systems Principles, Copper Mountain Resort, CO, USA", December 1995, p. 251–266.

[63] P. TH. EUGSTER, P. FELBER, R. GUERRAOUI, A.-M. KERMARREC. *The Many Faces of Publish/Subscribe*, in "ACM Computing Surveys", vol. 35, n° 2, June 2003, p. 114–131.

[64] A. FLISSI, P. MERLE. *Une démarche dirigée par les modèles pour construire les machines de déploiement des intergiciels à composants*, in "L'Objet", Hermès, vol. 11, n° 1-2, 2005, p. 79–94.

[65] J. O. KEPHART, D. M. CHESS. *The Vision of Autonomic Computing*, in "Computer - IEEE Computer Magazine", vol. 36, n° 1, 2003, p. 41–50.

[66] J. O. KEPHART. *Research Challenges of Autonomic Computing*, in "ICSE '05: Proceedings of the 27th international conference on Software engineering, New York, NY, USA", ACM Press, 2005, p. 15–22, http://doi.acm.org/10.1145/1062455.1062464.

[67] G. KICZALES, J. DES RIVIÈRES, D. BOBROW. *The Art of the Metaobject Protocol*, MIT Press, 1991.

[68] G. KICZALES. *Aspect-Oriented Programming*, in "ACM Computing Surveys", vol. 28, n° 4, 1996, 154, http://doi.acm.org/10.1145/242224.242420.

[69] G. MULLER, R. MARLET, E. VOLANSCHI, C. CONSEL, C. PU, A. GOEL. *Fast, Optimized Sun RPC Using Automatic Program Specialization*, in "Proceedings of the 18th International Conference on Distributed Computing Systems, Amsterdam, The Netherlands", IEEE Computer Society Press, May 1998, p. 240–249.

[70] D. OPPENHEIMER, A. GANAPATHI, D. A. PATTERSON. *Why do Internet services fail, and what can be done about it?*, in "4th USENIX Symposium on Internet Technologies and Systems (USITS '03)", March 2003.

[71] I. PIUMARTA, F. RICCARDI. *Optimizing direct threaded code by selective inlining*, in "1998 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'98), Montreal, Canada", 1998.

[72] C. SZYPERSKI. *Component Software - Beyond Object-Oriented Programming*, 2nd ed., 589 pp., Addison-Wesley, 2002.

[73] R. VAN RENESSE, K. P. BIRMAN, W. VOGELS. *Astrolabe: a Robust and Scalable Technology for Distributed Systems Monitoring, Management, and Data Mining* , in "ACM Transactions on Computer Systems", vol. 21, n^{o} 2, May 2003, p. 164–206.