INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team PARIS

Programming Parallel and Distributed Systems for Large Scale Numerical Simulation Applications

Rennes

THEME NUM

Activity Report

2006

Table of contents

1. Team

The PARIS *Project-Team was created at* IRISA *in December 1999. In November 2001, it has been established as a joint project-team (projet commun) between* IRISA *and the Brittany Campus of* ENS CACHAN. *Since, the project activity is jointly supervised by a ad-hoc Committee on an annual basis. Regarding 2005–2006, this committee met at* ENS CACHAN *on April 5, 2006, jointly with the similar committee for the DistribCom Project-Team, the other joint project-team between* IRISA *and the Brittany Campus of* ENS CACHAN.

Head of project-team

Thierry Priol [Research Director (DR) INRIA, HdR]

Administrative assistants

Maryse Auffray [Secretary (TR) INRIA]
Sandrine L'Hermitte [XTREEMOS Scientific Coordinator Assistant INRIA, since June 2006]
Päivi Palosaari [CoreGRID Scientific Coordinator Assistant INRIA]

Staff members Inria

Gabriel Antoniu [Research Associate (CR)]
Yvon Jégou [Research Associate (CR)]
David Margery [Research Engineer (IR, 50%)]
Christine Morin [Research Director (DR), HdR]
Christian Pérez [Research Associate (CR), HdR]

Staff members University Rennes 1

Françoise André [Professor, HdR]
Jean-Pierre Banâtre [Professor, HdR]
Pascal Morillon [Engineer]

Staff members Insa de Rennes

Jean-Louis Pazat [Professor, HdR]

Staff member Ens Cachan

Luc Bougé [Professor, ENS CACHAN Brittany Campus, HdR]

Project technical staff (temporary positions)

Landry Breuil [INRIA, ANR MD Project Lego]
Pascal Gallard [INRIA, COCA Project, till May 2006]
Raúl López Lozano [INRIA, ANR Project DiscoGrid]
Renaud Lottiaux [INRIA, COCA Project, till May 2006]
Jean Parpaillon [INRIA, Associate Engineer (IA)]
Oscar Sanchez [INRIA, XTREEMOS NoE, since September 2006]

PhD students

Jérémy Buisson [MENRT Grant (till September 2006)[1], then on a Post-Doc temporay ATER Grant]
Loïc Cudennec [INRIA and Brittany Regional Council Grant]
Boris Daix [CIFRE EDF industrial Grant (since February 2006)]
Matthieu Fertré [MENRT Grant, till June 2006]
Nagib Abi Fadel [SARIMA Grant]
Hinde Lilia Bouziane [INRIA Grant]
Mathieu Jan [INRIA and Brittany Regional Council Grant (till September 2006)[2]]
Emmanuel Jeanvoine [CIFRE EDF industrial Grant]
Sébastien Monnet [MENRT Grant (till September 2006)[3]]
Yann Radenac [MENRT Grant (till September 2006), then on a temporary ATER Grant]
Thomas Ropars [MENRT Grant (since October 2006)]

[1] PhD defended on September 25, 2006.
[2] PhD defended on November 20, 2006.
[3] PhD defended on November 30, 2006.

Post-doctoral fellow

Adrien Lèbre [INRIA Post-Doc (since October 2006)]
Louis Rilling [MENRT Grant (till August 2006)]
Xuanhua Shi [INRIA Post-Doc (since September 2006)]

2. Overall Objectives

2.1. General objectives

The PARIS Project-Team aims at contributing to the programming of parallel and distributed systems for large-scale numerical simulation applications. Its goal is to design operating systems and middleware to ease the use of such computing infrastructure for the targeted applications. Such applications enable the speed-up of the design of complex manufactured products, such as cars or aircrafts, thanks to numerical simulation techniques.

As computer performance rapidly increases, it is possible to foresee in the near future comprehensive simulations of these designs that encompass multi-disciplinary aspects (structural mechanics, computational fluid dynamics, electromagnetism, noise analysis, etc.). Numerical simulations of these different aspects will not be carried out by a single computer due to the lack of computing and memory resources. Instead, several clusters of inexpensive PCs, and probably federations of clusters (aka. *Grids*), will have to be simultaneously used to keep simulation times within reasonable bounds. Moreover, simulation will have to be performed by different research teams, each of them contributing its own simulation code. These teams may all belong to a single company, or to different companies possessing appropriate skills and computing resources, thus adding geographical constraints. By their very nature, such applications will require the use of a computing infrastructure that is *both* parallel and distributed.

The PARIS Project-Team is engaged in research along five topics: *Operating System and Runtime for Clusters and Grids*, *Middleware for Computational Grids*, *Large-scale Data Management for Grids*, *Advanced Models for the Grid* and *Experimental Grid Infrastructures*.

Topic *P2P System Foundations*, that was described in the previous activity report, has been spinned-off to a new project-team, called *ASAP*, headed by Anne-Marie Kermarrec, a former member of the PARIS Project-Team.

The research activities of the PARIS Project-Team encompass both basic research, seeking conceptual advances, and applied research, to validate the proposed concepts against *real* applications. The project-team is also heavily involved in managing a national grid computing infrastructure (GRID 5000) enabling large-scale experiments.

2.1.1. Parallel processing to go faster

Given the significant increase of the performance of microprocessors, computer architectures and networks, clusters of standard personal computers now provide the level of performance to make numerical simulation a handy tool. This tool should not be used by researchers only, but also by a large number of engineers, designing complex physical systems. Simulation of mechanical structures, fluid dynamics or wave propagation can nowadays be carried out in a couple of hours. This is made possible by exploiting multi-level parallelism, simultaneously at a fine grain within a microprocessor, at a medium grain within a single multi-processor PC, and/or at a coarse grain within a cluster of such PCs. This unprecedented level of performance definitely makes numerical simulation available for a larger number of users such as SMEs. It also generates new needs and demands for more accurate numerical simulation. Traditional parallel processing alone cannot meet this demand.

2.1.2. Distributed processing to go larger

These new needs and demands are motivated by the constraints imposed by a worldwide economy: making things faster, better and cheaper.

2.1.2.1. Large-scale numerical simulation.

Large scale numerical simulation will without a doubt become one of the key technologies to meet such constraints. In traditional numerical simulation, only one simulation code is executed. In contrast, it is now required to *couple* several such codes together in a single simulation.

A large-scale numerical simulation application is typically composed of several codes, not only to simulate one physics, but to perform multi-physics simulation. One can imagine that the simulation times will be in the order of weeks and sometimes months depending on the number of physics involved in the simulation, and depending on the available computing resources.

Parallel processing extends the number of computing resources locally: it cannot significantly reduce simulation times, since the simulation codes will not be localized in a single geographical location. This is particularly true with the global economy, where complex products (such as cars, aircrafts, etc.) are not designed by a single company, but by several of them, through the use of subcontractors. Each of these companies brings its own expertise and tools such as numerical simulation codes, and even its private computing resources. Moreover, they are reluctant to give access to their tools as they may at the same time compete for some other projects. It is thus clear that distributed processing cannot be avoided to manage large-scale numerical applications

2.1.2.2. Resource aggregation.

More generally, the development of large scale distributed systems and applications now rely on resource sharing and aggregation. Distributed resources, whether related to computing, storage or bandwidth, are aggregated and made available to the whole system. Not only this aggregation greatly improves the performance as the system size increases, but many applications would simply not have been possible without such a model (peer-to-peer file sharing, ad-hoc networks, application-level multicast, publish-subscribe applications, etc.).

2.1.3. Scientific challenges of the Paris Project-Team

The design of large-scale simulation applications raises technical and scientific challenges, both in applied mathematics and computer science. The PARIS Project-Team mainly focuses its effort on Computer Science. It investigates new approaches to build software mechanisms that hide the complexity of programming computing infrastructures that are *both* parallel and distributed. Our contribution to the field can thus be summarized as follows:

combining parallel and distributed processing whilst preserving performance and transparency.

This contribution is developed along five directions.

Operating system and runtime for clusters and grids. The challenge is to design and build an operating system for clusters hiding to the programmers and the users, the fact that resources (processors, memories, disks) are distributed. A PC cluster with such an operating system looks like a traditional multi-processor running a Single System Image (SSI).

Middleware for computational grids. The challenge is to design a middleware implementing a component-based approach for grids. Large-scale numerical applications will be designed by combining together a set of components encapsulating simulation codes. The challenge is to seamlessly mix both parallel and distributed processing.

Large-scale data management for grids. One of the key challenges in programming grid computing infrastructures for real, is data management. It has to be carried out at an unprecedented scale, and to cope with the native dynamicity and heterogeneity of the underlying grids.

Advanced models for the Grid. This topic aims at contributing to study unconventional approaches for the programming of grids based on the *chemical metaphors*. The challenge is to exploit such metaphors to make the use, including the programming, of grids more intuitive and simpler.

Experimental Grid Infrastructure. The challenge here is to be able to design and to build an *instrument* (in the sense of a large scientific instrument, like a telescope) for computer scientists involved in grid research. Such an instrument has to be highly reconfigurable and scalable to several thousand of resources.

2.2. Operating system and runtime for clusters and grids

Clusters, made up of homogeneous computers interconnected via high-performance networks, are now widely used as general-purpose, high-performance computing platforms for scientific computing. While such an architecture is attractive with respect to its price/performance ratio, there still exists a large potential for efficiency improvement at the software level. System software can be improved to better exploit cluster hardware resources. Programming environments need to be developed with both the cluster and human programmer efficiency in mind.

We believe that cluster programming remains difficult. This is due to the fact that clusters suffer from a lack of dedicated operating system providing a single system image (SSI). A single system image provides the illusion of a single, powerful and highly-available computer to cluster users and programmers, as opposed to a set of independent computers, whose resources have to be managed locally.

Several attempts to build an SSI have been made at the middleware level as Beowulf [94], PVM [80] or MPI [89]. However, these environments only provide a *partial* SSI. Our approach in the PARIS Project-Team is to design and implement a *full* SSI in the operating system. Our objective is to combine ease of use, high performance and high availability. *All* physical resources (processor, memory, disk, etc.) and kernel resources (process, memory pages, data streams, files, etc.) need to be visible and accessible from *all* cluster nodes. Cluster reconfigurations due to a node addition, eviction or failure, need to be automatically dealt with by the system, transparently to the applications. Our SSI operating system (SSI OS) is designed to perform global, dynamic and integrated resource management.

As the execution time of scientific applications may be larger than the cluster mean time between failures, checkpoint/restart facilities need to be provided, not only for sequential applications but also for parallel applications. This is independent of the underlying communication paradigm. Even though backward error recovery (BER) has been extensively studied from the theoretical point of view, an efficient implementation of BER protocols, transparent to the applications, is still a research challenge. There are very few implementations of recovery schemes for parallel applications. Our approach is to identify and implement as part of the SSI OS, a set of building blocks that can be combined to implement various checkpointing strategies and their optimization for parallel applications, whatever inter-process communication (IPC) layer they use.

In addition to our research activity on operating system, we also study the design of runtimes for supporting parallel languages on clusters. A runtime is a software offering services dedicated to the execution of a particular language. Its objective is to tailor the general system mechanisms (memory management, communication, task scheduling, etc.) to achieve the best performance given the target machine and its operating system. The main originality of our approach is to use the concept of *distributed shared memory* (DSM) as the basic communication mechanism within the runtime. We are essentially interested in Fortran and its OpenMP extensions [70]. The Fortran language is traditionally used in the simulation applications we focus on. Our work is based on the operating system mechanisms studied in the PARIS Project-Team. In particular, the execution of OpenMP programs on a cluster requires a global address space shared by threads deployed on different cluster nodes. We rely on the two distributed shared memory systems we have designed: one at user level, implementing weak memory consistency models, and the other one at operating-system level, implementing the sequential consistency model.

2.3. Middleware systems for computational grids

Computational grids are very powerful machines as they aggregate huge computational resources. A lot of work has been carried out with respect to grid resource management. Existing grid middleware systems mainly focus on resource management like discovery, registration, security, scheduling, etc. However, they provide very little support for grid-oriented programming models.

A suitable grid programming model should be able to take into account the dual nature of a computational grid which is a distributed set of (mainly) parallel resources.

Our general objective is to propose such a programming model and to provide adequate middleware systems. Distributed object or component models seems to be a promising solution. However, they need to be tailored for scientific applications. In particular, the parallel applications have to be encapsulated into objects or components. New paradigms of communication between *parallel* objects or components have to be designed, together with the required runtime support, deployment facilities, and capacity for dynamic adaptability.

The first issue is the relationship between object or component models, which should handle the distributed nature of grid, and the parallelism of computational codes, which should take into account the parallelism of resources. It is thus required to efficiently integrate both worlds into a coherent, single vision.

The second issue concerns the simplicity and the scalability of communication between parallel codes. As the available bandwidth is larger than what a single resource could consume, parallel communication flows should allow a more efficient utilization of network resources. Advanced flow control should be used to avoid congesting networks. A crucial aspect of this issue is the support for data redistribution involved in the communication between parallel codes.

The third issue refers to the dynamic behavior of applications. While software component models are demonstrating their usefulness in capturing the static architecture of applications, there are still few results on how to deal with the dynamic aspects. The composition operator should be revised so as not to hide such dynamic aspects into the component implementation code.

Promoting a programming model that simultaneously supports distributed as well as parallel middleware systems, independently of the actual resources, raises three new issues. First, middleware systems should be decoupled from the actual networks so as to be deployed on any kind of network. Second, several middleware systems should be able to be *simultaneously* active within a same process. Third, the solutions to the two previous issues should meet the user requirements for high performance.

The deployment of applications is another issue. Not only is it important to specify the deployment in term of the computational resources (GFlop/s, amount of memory, etc.), but it is also crucial to specify the requirements related to communication resources, such as the amount of bandwidth, or the latency between computational resources. Moreover, we have to deal with applications integrating several distributed middleware systems, like MPI, CORBA, JXTA, etc.

The last issue deals with the dynamic nature of computational grids. As targeted applications may run for very long time, the grid environment is expected to change. Not only middleware systems should support adaptability, but they should also be able to detect variations and to self-adapt. For example, it should be possible to partially redeploy an application on the fly, to benefit from new resources.

2.4. Large-scale data management for grids

A major contribution of the grid computing environments developed so far is to have decoupled *computation* from *deployment*. Deployment is typically considered as an *external service* provided by the underlying infrastructure, in charge of locating and interacting with the physical resources. In contrast, as of today, no such sophisticated service exists regarding *data management* on the grid: the user is still left to explicitly store and transfer the data needed by the computation between these sites. Like deployment, we claim that an adequate approach to this problem consists in decoupling *data management* from *computation*, through an *external service* tailored to the requirements of scientific applications. We focus on the case of a grid consisting of a federation of distributed clusters. Such a *data sharing service* should meet two main properties: *persistence* and *transparency*.

First, the data sets used by the grid computing applications may be very large. Their transfer from one site to another may be costly (in terms of both bandwidth and latency), so that such data movements should be carefully optimized. Therefore, the data management service should allow data to be *persistently* stored on the grid infrastructure independently of the applications, in order to allow their reuse in an efficient way.

Second, a data management service should provide *transparent* access to data. It should handle data localization and transfer without any help from the programmer. Yet, it should make good use of additional information and hints provided by the programmer, if any. The service should also transparently use adequate replication strategies and consistency protocols to ensure data availability and consistency in a large-scale, dynamic architecture.

Given that our target architecture is a federation of clusters, several additional constraints need to be addressed. The clusters which make up the grid are not guaranteed to remain available constantly. Nodes may leave due to technical problems or because some resources become temporarily unavailable. This should obviously not result in disabling the data management service. Also, new nodes may dynamically join the physical infrastructure: the service should be able to dynamically take into account the additional resources they provide. Therefore, adequate strategies need to be set up in order for the service to efficiently interact with the resource management system of the grid.

On the other hand, it should be noted that the algorithms proposed for parallel computing have often been studied on small-scale configurations. Our target architecture is typically made of thousands of computing nodes, say tens of hundred-node clusters. It is well-known that designing low-level, explicit MPI programs is most difficult at such a scale. In contrast, peer-to-peer approaches have proved to remain effective at a large scale, and can serve as fruitful inspiration sources.

Finally, data is generally shared in grid applications, and can be modified by multiple partners. Traditional replication and consistency protocols designed for DSM systems have often made the assumption of a small-scale, static, homogeneous architecture. These hypotheses need to be revisited and this should lead to new consistency models and protocols adapted to a dynamic, large-scale, heterogeneous architecture.

2.5. Advanced programming models for the Grid

Till now, research activities related to the grid have focused on the design and implementation of middleware and tools to experiment grid infrastructure with applications. Little attention has been paid to programming models suitable for such widely computing infrastructures. Programming such infrastructures is still done at a very low level. This situation may somehow be compared to using assembly language to program complex processors. Our objective is to study approaches for grid programming that do not expose the architectural details of the computing infrastructure to the programmers. More specifically, we are considering unconventional approach based on the *chemical reaction* paradigm, and more precisely the GAMMA Model [76].

GAMMA is based on multiset rewriting. The unique data structure in GAMMA is the multiset (a set than can contain several occurrences of the same element), which can be seen as a *chemical solution*. A simple program is a set of rules $Reaction\ \ condition \rightarrow Action$. Execution proceeds, without any explicit order, by replacing elements in the multiset satisfying the reaction condition by the products of the action (*chemical reaction*). The result is obtained when a stable state is reached, that is, when no more reactions applies. Our objective is to express the coordination of Grid components or services through a set of rules, while the multiset represents the services that have to be coordinated.

2.6. Experimental Grid Infrastructures

The PARIS Project-Team is engaged in research along five research topics: *Operating System and Runtime for Clusters and Grids*, *Middleware for Computational Grids*, *Large-scale Data Management for Grids*, *Advanced Models for the Grid* and *Experimental Grid Infrastructures*. The concepts proposed by each of these topics must be validated against real applications on realistic hardware. The project-team manages a computation platform dedicated to operating system and middleware experimentations. This platform is integrated within GRID 5000, a national computing infrastructure dedicated to large-scale Grid and peer-to-peer experiments. The GRID 5000 infrastructure federates experimental platforms (currently 9 platforms) across France. These platforms are connected through Renater using dedicated Gigabit Ethernet links.

Our experimental platform is heterogeneous: PowerPC and PC families of processors, 32-bit and 64-bit architectures, Linux and Mac OS X operating systems. Various high-performance interconnection technologies such as Myrinet and InfiniBand are available on groups of nodes. Heterogeneity allows realistic validation of interoperability of middleware and P2P systems. On the other hand, our platform is composed of sufficiently large groups of homogeneous computation nodes: 66 dual Xeon, 166 dual Opterons, 33 Xserve G5. This allows to evaluate the scalability of operating systems, runtimes and applications on various architectures.

Our experimental platform is dedicated to operating system and middleware experimentation: it is possible to repeat experiments in the same environment (same machines, same network, etc.). The allocation of the resources to the experiments is handled through *GridPrems*, a collaborative resource manager developed in our group and through *OAR*, a job manager developed by the Grenoble partner group of GRID 5000.

3. Scientific Foundations

3.1. Introduction

Research activity within the PARIS Project-Team encompasses several areas: operating systems, middleware and programming models. We have chosen to provide a brief presentation of some of the scientific foundations associated with them.

3.2. Data consistency

A shared virtual memory system provides a global address space for a system where each processor has only physical access to its local memory. Implementating of such a concept relies on the use of complex cache coherence protocols to enforce data consistency. To allow the correct execution of a parallel program, it is required that a read access performed by one processor returns the value of the last write operation previously performed by any other processor. Within a distributed or parallel a system, the notion of the *last* memory access is sometimes partially defined only, since there is no global clock to provide a total order of the memory operation.

It has always been a challenge to design a shared virtual memory system for parallel or distributed computers with distributed physical memories, capable of providing comparable performance with other communication models such as message-passing. *Sequential Consistency* [86] is an example of a memory model for which all memory operations are consistent with a total order. Sequential Consistency requires that a parallel system having a global address space appears to be a multiprogramming uniprocessor system to any program running on it. Such a strict definition impacts on the performance of shared virtual memory systems due to the large number of messages that are required (page access, invalidation, control, etc.). Moreover Sequential Consistency is not necessarily required to correctly run parallel programs, in which memory operations to the global address space are guarded by synchronization primitives.

Several other memory models have thus been proposed to relax the requirements imposed by sequential consistency. Among them, *Release Consistency* [81] has been thoroughly studied since it is well adapted to programming parallel scientific applications. The principle behind Release Consistency is that memory accesses are (should?) always be guarded by synchronization operations (locks, barriers, etc.), so that the shared memory system only needs to ensure consistency at synchronization points. Release Consistency requires the use of two new operations: *acquire* and *release*. The aim of these two operations is to specify when to propagate the modifications made to the shared memory systems. Several implementations of Release Consistency have been proposed [84]: an *eager* one, for which modifications are propagated at the time of a release operation; and a *lazy* one, for which modifications are propagated at the time of an acquire operation. These alternative implementations differ in the number of messages that needs to be sent/received, and in the complexity of their implementation [85].

Implementations of Release Consistency rely on the use of a logical clock such as a vector clock [88]. One of the drawback of such a logical clock is its lack of scalability when the number of processors increases, since the vector carries one entry per processor. In the context of computing systems that are both parallel and distributed, such as a grid infrastructure, the use of a vector clock is impossible in practice. It is thus necessary to find new approaches based on logical clocks that do not depend on the number of processors accessing the shared memory system. Moreover, these infrastructures are natively *hierarchical*, so that the consistency model should better take advantage of it.

3.3. High availability

> “A distributed system is one that stops you getting any work done when a machine you’ve never even heard about crashes.” (Leslie Lamport)

The *availability* [82] of a system measures the ratio of service accomplishment conforming to its specifications, with respect to elapsed time. A system *fails* when it does not behave in a manner consistent with its specifications. An error is the consequence of a *fault* when the faulty part of the system is activated. It may lead to the system *failure*. In order to provide highly-available systems, *fault tolerance techniques* [87] based on redundancy can be implemented. Abstractions like *group membership*, *atomic multicast*, *consensus*, etc. have been defined for fault-tolerant distributed systems.

Error detection is the first step in any fault tolerance strategy. *Error treatment* aims at avoiding that the error leads to the system failure.

Fault treatment consists in avoiding that the fault be activated again. Two classes of techniques can be used for fault treatment: *reparation* which consists in eliminating or replacing the faulty module; and *reconfiguration* which consists in transferring the load of the faulty element to valid components.

Error treatment can be of two forms: *error masking* or *error recovery*. Error masking is based on hardware or software redundancy in order to allow the system to deliver its service despite the error. Error recovery consists in restoring a correct system state from an erroneous state. In *forward error recovery* techniques, the erroneous state is transformed into a safe state. *Backward error recovery* consists in periodically saving the system state, called a *checkpoint*, and rolling back to the last saved state if an error is detected.

A *stable storage* guarantees three properties in presence of failures: (1) *integrity*, data stored in stable storage is not altered by failures; (2) *accessibility*, data stored in stable storage remains accessible despite failures; (3) *atomicity*, updating data stored in stable storage is an all or nothing operation. In the event of a failure during the update of a group of data stored in stable storage, either all data remain in their initial state or they all take their new value.

3.4. Distributed data management

Past research on distributed data management led to three main approaches. Currently, the most widely-used approach to data management for distributed grid computation relies on *explicit data transfers* between clients and computing servers. As an example, the *Globus* [68] platform provides data access mechanisms (like data catalogs) based on the *GridFTP* protocol. Other explicit approaches (e.g., *IBP*) provide a large-scale data storage system, consisting of a set of buffers distributed over Internet. The user can “rent” these storage areas for efficient data transfers.

In contrast, *Distributed Shared Memory* (DSM) systems provide *transparent* data sharing, via a virtual, unique address space accessible to physically distributed machines. It is the responsibility of the DSM system to localize, transfer, replicate data, and guarantee their consistency according to some semantics. Within this context, a variety of consistency models and protocols have been defined. Nevertheless, existing DSM systems have generally shown satisfactory efficiency only on small-scale configurations, up to a few tens of nodes.

Recently, *peer-to-peer* (P2P) has proven to be an efficient approach for large-scale resource (data or computing resources) sharing [90]. The peer-to-peer communication model relies on a symmetric relationship between peers which may act both as clients and servers. Such systems have proven able to manage very large and dynamic configurations (millions of peers). However, several challenges remain. More specifically, as far as data sharing is concerned, most P2P systems focus on sharing *read-only* data, that do not require data consistency management. Some approaches, like *OceanStore* and *Ivy*, deal with *mutable* data in a P2P with restricted use. Today, one major challenge in the context of large-scale, distributed data management is to define appropriate models and protocols allowing to guarantee both *consistency* of replicated data and *fault tolerance*, in *large-scale, dynamic environments*.

3.5. Component model

Software component technology [96] has been emerging for some years, even though its underlying intuition is not very recent. Building an application based on components emphasizes programming by *assembly*, that is, *manufacturing*, rather than by *development*. The goals are to focus expertise on domain fields, to improve software quality, and to decrease the time-to-market thanks to reuse of existing codes.

The CORBA Component Model (CCM), which is part of the latest CORBA [92] specifications (Version 3), appears to be the most complete specification for components. It allows the deployment of a set of components into a distributed environment. Moreover, it supports heterogeneity of programming languages, operating systems, processors, and it also guarantees interoperability between different implementations. However, CCM does not provide any support for parallel components.

The CORBA Component Architecture (CCA) Forum [72] aims at developing a standard which specifically addresses the needs of the HPC community. Its objective is to define a minimal set of standard interfaces that any high-performance component framework should provide to components, and may expect from them, in order to allow disparate components to be composed together into a running application. CCA aims at supporting *both* parallel and distributed applications.

3.6. Adaptability

Due to the dynamic nature of large-scale distributed systems in general, and the Grid in particular, it is very hard to design an application that fits well in any configuration. Moreover, constraints such as the number of available processors, their respective load, the available memory and network bandwidth are not static. For these reasons, it is highly desirable that an application could take into account this dynamic context in order to get as much performance as possible from the computing environment.

Dynamic adaptation of a program is the modification of its behavior according to changes of the environment. This adaptivity can be achieved in many different ways, ranging from a simple modification of some parameters, to the total replacement of the running code. In order to achieve adaptivity, a program needs to be able to get information about the environment state, to make a decision according to some optimization rules, and to modify or replace some parts of its code.

Adaptivity has been implemented by designing ad hoc applications that take into account the specificities of the target environment. For example, this was done for the Web applications access protocol on mobile networks by defining the WAP protocol [71]. A more general way is to provide mechanisms enabling dynamic self-adaptivity by changing the program's behavior. In most cases, this has been achieved by embedding the adaptation mechanism within the application code. For example, the AdOC compression algorithm [83] includes such a mechanism to dynamically change the compression level according to the available resources.

However, it is desirable to separate the adaptation engine from the application code, in order to make the code easier to maintain, and to easily change or improve the adaptation policy. This was done for wireless and mobile environments by implementing a framework [78] that provides generic mechanisms for the adaptation process, and for the definition of the adaptation rules.

3.7. Chemical programming

The chemical reaction metaphor has been discussed in various occasions in the literature. This metaphor describes computation in terms of a chemical solution in which molecules (representing data) interact freely according to reaction rules. Chemical models use the multiset as their basic data structure. Computation proceeds by rewritings of the multiset which consume elements according to reaction conditions and produce new elements according to specific transformation rules.

To the best of our knowledge, the GAMMA formalism was the first "chemical model of computation" proposed as early as in 1986 [75] and extended later [76].

A GAMMA program is a collection of reaction rules acting on a multiset of basic elements. A reaction rule is made of a condition and an action. Execution proceeds by replacing elements satisfying the reaction condition by the elements specified by the action. The result of a GAMMA program is obtained when a stable state is reached that is to say when no more reactions can take place. Here is an example illustrating the GAMMA style of programming:

$$primes = \mathbf{replace} x, y \mathbf{by} y \mathbf{if} multiple(x, y)$$

The reaction $primes$ computes the prime numbers lower or equal to a given number N when applied to the multiset of all numbers between 2 and N ($multiple(x, y)$ is true if and only if x is a multiple of y). Let us emphasize the conciseness and elegance of these programs. Nothing had to be said about the order of evaluation of the reactions. If several disjoint pairs of elements satisfy the condition, the reactions can be performed in parallel.

GAMMA makes it possible to express programs without artificial sequentiality. By artificial, we mean sequentiality only imposed by the computation model and unrelated to the logic of the program. This allows the programmer to describe programs in a very abstract way. In some sense, one can say that GAMMA programs express the very idea of an algorithm without any unnecessary linguistic idiosyncrasies. The interested reader may find in [76] a long series of examples (string processing problems, graph problems, geometry problems, etc.) illustrating the GAMMA style of programming and in [74] a review of contributions related to the chemical reaction model. Later, the idea was developed further into the CHAM [77], the P-systems [93], etc. Although built on the same basic paradigm, these proposals have different properties and different expressive powers.

The γ-calculus [73] is an attempt to identify the basic principles behind chemical models. It exhibit a minimal chemical calculus, from which all other "chemical models" can be obtained by addition of well-chosen features. Essentially, this minimal calculus incorporates the γ-reduction which expresses the very essence of the chemical reaction, and the associativity and commutativity rules which express the basic properties of chemical solutions.

4. Application Domains

4.1. Application Domains

Keywords: *Scientific computing*, *co-operative applications*, *large-scale computing*.

The project-team research activities address scientific computing and specifically numerical applications that require the execution of several codes simultaneously. This kind of applications requires both the use of parallel and distributed systems. Parallel processing is required to address performance issues. Distributed processing is needed to fulfill the constraints imposed by the localization and the availability of resources, or for confidentiality reasons. Such applications are being experimented within contracts with the industry or through our participation to application-oriented research grants.

5. Software

5.1. Kerrighed

Keywords: *Cluster operating system*, *checkpointing*, *co-operative caching*, *distributed shared memory (DSM)*, *distributed file system*, *global scheduling*, *high availability*, *process migration*, *single system image (SSI)*.

Participants: Matthieu Fertré, Pascal Gallard, Renaud Lottiaux, Christine Morin, Jean Parpaillon.

Contact: Christine Morin, `Christine.Morin@irisa.fr`

URL: http://www.kerrighed.org/ and http://ssi-oscar.gforge.inria.fr/

Status: Registered at APP, under Reference `IDDN.FR.001.480003.006.S.A.2000.000.10600`.

License: GNU General Public License version 2. KERRIGHED is a registered trademark.

Presentation: KERRIGHED is a *Single System Image* (SSI) operating system for high-performance computing on clusters. It provides the user with the illusion that a cluster is a virtual SMP machine.

In KERRIGHED, all resources (processes, memory segments, files, data streams) are globally and dynamically managed to achieve the SSI properties. Global resource management makes distribution of resources transparent throughout the cluster nodes, and allows to take advantage of the whole cluster hardware resources for demanding applications. Dynamic resource management enables transparent cluster reconfigurations (node addition or eviction) for the applications, and high availability in the event of node failures. In addition, a checkpointing mechanism is provided by KERRIGHED to avoid restarting applications from the beginning when some node failure occurs.

KERRIGHED preserves the interface of a standard, single-node operating system, which is familiar to programmers. Legacy sequential or parallel applications running on this standard operating system can be executed without modification on top of KERRIGHED, and further optimized if needed.

KERRIGHED is not an entirely new operating system developed from scratch. Just in the opposite, it has been designed and implemented as an extension to an existing standard operating system. KERRIGHED only addresses the distributed nature of the cluster, while the native operating system running on each node remains responsible for the management of local physical resources. Our current prototype is based on *Linux*, which is extended using the standard module mechanism. The Linux kernel itself has only been slightly modified.

A public mailing list (`kerrighed.users@irisa.fr`) and a technical forum are available to provide a support to KERRIGHED users.

Current status: KERRIGHED (version V1.0.2) includes 70,000 lines of code (mostly in C). It involved more than 200 persons-months. It provides a customizable, cluster-wide process scheduler, a cluster-wide Unix process interface, high-performance stream migration allowing migration of MPI processes, process checkpointing, and an efficient distributed file system. It also offers a complete *Pthread* support, allowing to execute legacy OpenMP and multithreaded applications on a cluster without any recompilation. KERRIGHED SSI features are customizable.

A live-CD of KERRIGHED based on *Knoppix* is also available. It eases KERRIGHED installation for demonstrations or for evaluation, by users not familiar with Linux installation process.

In 2006, KERRIGHED has been ported to Linux 2.6.11. The code has significantly been improved during this port resulting in a more compact software. Moreover, KERRIGHED is also distributed as an *official* spin-off OSCAR package with the SSI-OSCAR package. Since November 2006, SSI-OSCAR packages based on the development version of KERRIGHED and OSCAR 5.0, are available for Linux distributions supported by OSCAR (e.g., Fedora Core 5, RedHat Enterprise Linux 4, etc.). A port to the Debian Linux distribution has also been carried out.

Demonstrations of KERRIGHED have been presented in 2006 at *Linux Expo* (Paris, February 2006, Pascal Gallard, Renaud Lottiaux, Jean Parpaillon and Christine Morin), and *Supercomputing 2006 Conference* (Tampa, Florida, November 2006, Jean Parpaillon). KERRIGHED has also been presented by Jean Parpaillon at the *Paris Capitale du Libre* event in Paris, in June 2006.

5.2. PadicoTM

Keywords: *Grid*, *communication framework*, *middleware system*.

Participants: Christian Pérez, Thierry Priol.

Contact: Christian Pérez, `Christian.Perez@irisa.fr`

URL: http://runtime.futurs.inria.fr/PadicoTM/

Status: Registered at APP, under Reference `IDDN.FR.001.260013.000.S.P.2002.000.10000`.

License: GNU General Public License version 2.

Presentation: PADICOTM is an open integration framework for communication middleware and runtime systems. It enables several middleware systems (such as CORBA, MPI, SOAP, etc.) to be used at the same time. It provides an efficient and transparent access to all available networks with the appropriate method.

PADICOTM is composed of a core, which provides a high-performance framework for networking and multi-threading, and services, plugged into the core. High-performance communications and threads are obtained thanks to MARCEL and MADELEINE, provided by PM 2 . The PADICOTM core aims at making the different services running at the same time run in a co-operative way rather than competitive.

An extended set of commands is provided with PADICOTM to ease the compilation of its modules (`padico-cc`, `padico-c++`, etc.). In particular, it hides the differences between the various implementations of CORBA.

PadicoControl is a Java application that helps to control the deployment of PADICOTM application. It allows a user to select the deployment node, and to perform individual or collective operations like loading or running a PADICOTM module.

PadicoModule (still under development) is a Java application which assists the low-level administration of a PADICOTM installation. It allows to check module dependency, to modify module attributes, etc. It can work on the local file system as well as through a network, thanks to a SOAP daemon being part of the service.

The repository of project is handled by the INRIA Gforge within the Padico project. A public mailing list (`padico-users@listes.irisa.fr`) is available to support users of PADICOTM.

Current status: The development of PADICOTM has started at the end of 2000. It involved around 100 persons-months.

The latest release of PADICOTM is Version 0.4 (November 2006). It includes the PADICOTM core, *PadicoControl*, *myCORBA* and includes external software: a customized version of PM 2 and a regular version of *Expat* (1.95.2). One major feature of this version is that is does not require any special version of the supported middleware systems. Current supported middleware systems are *omniORB3*, *omniORB4* and *Mico* 2.3.x for CORBA, MPICH 1.1.2 and MPICH 1.2.5 for MPI, and *gSOAP* 2.6.x for SOAP.

PADICOTM has been funded by the RMI Project of the French ACI GRID program . As we are aware of, it has been used by several French projects: ACI GRID HydroGrid, ACI GRID EPSN, RNTL VTHD ++ and INRIA ARC RedGrid. It was also used in the European FET project POP. It is currently used by three French ANR CI projects: LEGO, DISC and NUMASIS.

5.3. PaCO++

Keywords: *CORBA*, *Grid*, *data parallelism*, *middleware system*.

Participants: Raúl López Lozano, Christian Pérez, Thierry Priol.

Contact: Christian Pérez, `Christian.Perez@irisa.fr`

URL: http://www.irisa.fr/paris/Paco++/

Status: Registered at APP, under Reference `IDDN.FR.001.450014.000.S.P.2004.000.10400`.

License: GNU General Public License version 2 and GNU Lesser General Public License version 2.1.

Presentation: The PACO++ objectives are to allow a simple and efficient embedding of a SPMD code into a parallel CORBA object and to allow parallel communication flows and data redistribution during an operation invocation on such a parallel CORBA object.

PACO++ provides an implementation of the concept of parallel object applied to CORBA. A parallel object is an object whose execution model is parallel. It is externally accessible through an object reference, whose interpretation is identical to a standard CORBA object.

PACO++ extends CORBA, but does not modify the underlying model. It is meant to be a *portable* extension to CORBA, so that it can be added to any CORBA implementation. The parallelism of an object is in fact considered to be an implementation feature of this object, and the OMG IDL is not dependent on it.

PACO++ is made of two components: a compiler and a runtime library.

The compiler generates parallel CORBA stub and skeleton from an IDL file which describes the CORBA interface, and from an XML file which describes the parallelism of the interface. The compilation is done in two steps. The first step involves a Java IDL-to-IDL compiler based on *SableCC*, a compiler of compiler, and *Xerces* for the XML parser. The second part, written in Python, generates the stubs files from templates configured with inputs generated during the first step.

The runtime, currently written in C++, deals with the parallelism of the parallel CORBA object. It is very portable thanks to the utilization of abstract APIs for communications, threads and redistribution libraries.

Current status: The development of PACO++ started at the end of 2002. It involved 60 persons-months. The first public version, referenced as PACO++ 0.1 has been released in November 2004. The second version (0.2) has been released in March 2005. It has been successfully tested on top of three CORBA implementations: *Mico*, *omniORB3* and *omniORB4*. Moreover, it supports PADICOTM.

The version 0.2 of PACO++ includes 7,000 lines of Java (around 250 kB), 5,000 lines of Python (around 390 kB), 14,000 lines of C++ (around 390 kB) and 2,000 lines of `shell`, `make` and `configure` scripts (60 kB).

PACO++ has been supported by the RMI Project of the French ACI GRID program. It has been used or it is used by several other French projects: ACI GRID HydroGrid, ACI GRID EPSN, RNTL VTHD ++ and INRIA ARC RedGrid. It is currently used within two French ANR CI projects: DISC and NUMASIS.

5.4. Adage

Keywords: *Grid*, *deployment*, *middleware system*.

Participants: Landry Breuil, Loïc Cudennec, Boris Daix, Mathieu Jan, Christian Pérez, Thierry Priol.

Contact: Christian Pérez, `Christian.Perez@irisa.fr`

URL: http://www.irisa.fr/paris/ADAGE/

Status: Under development.

License: GNU General Public License version 2.

Presentation: ADAGE (*Automatic Deployment of Applications in a Grid Environment*) is a research prototype that aims at studying the deployment issues related to multi-middleware applications. Its original contribution is to use a *generic* application description model (*GADe*) to transparently handle various middleware systems.

With respect to application submission, ADAGE requires an application description, which is specific to a programming model, a reference to a resource information service (MDS2, or an XML file), and a control parameter file. The application description is internally translated into a generic description, so as to support multi-middleware applications. The control parameter file allows a user to express constraints on the placement policy, which is specific to an execution. For example, a constraint may specify the latency and the bandwidth between a computational component and a visualization component.

The support of multi-middleware applications is based on a plug-in mechanism. The plug-in is involved in the conversion from the specific to the generic application description, but also during the execution phase so as to deal with specific middleware configuration actions.

ADAGE currently deploys static applications only. It supports standard programming models like MPI (*MPICH1-P4* and *MPICH-G2*), CCM and JXTA, as well as more advanced programming models like GRIDCCM. The current support of GRIDCCM is restricted to MPI-based parallel components.

Current status: The current (unstable) version of ADAGE includes 35,000 lines of C++. A non-public version has been used within the ACI GRID HydroGrid project. Current non-public versions are used within the ANR CI projects LEGO and DISC.

5.5. JuxMem

Keywords: *JXTA*, *Peer-to-peer*, *data grids*, *large-scale data management*.

Participants: Gabriel Antoniu, Luc Bougé, Landry Breuil, Loïc Cudennec, Mathieu Jan, Sébastien Monnet.

Contact: Gabriel Antoniu, `Gabriel.Antoniu@irisa.fr`

URL: http://juxmem.gforge.inria.fr/

License: GNU Lesser General Public License version 2.1.

Status: Registered at APP, under Reference `IDDN.FR.001.180015.000.S.P.2005.000.10000`.

Presentation: JUXMEM is a supportive platform for a data-sharing service for grid computing. This service addresses the problem of managing mutable data on dynamic, large-scale configurations. It can be seen as a hybrid system combining the benefits of *Distributed Shared Memory* (DSM) systems (transparent access to data, consistency protocols) and *Peer-to-Peer* (P2P) systems (high scalability, support for resource volatility). The target applications are numerical simulations, based on code coupling, with significant requirements in terms of data storage and sharing. JUXMEM's architecture decouples fault-tolerance management from consistency management. Multiple consistency protocols can be built using fault-tolerant building blocks such as *consensus*, *atomic multicast*, *group membership*. Currently, a hierarchical protocol implementing the entry consistency model is available. A more relaxed consistency protocol adapted to visualization is also available. A more detailed description of the approach is given in 6.4.1.

Current status: Two implementations are in progress, in Java and C. JuxMem is based on the *JXTA* generic platform for P2P services (Sun Microsystems, http://www.jxta.org/). At this time, it includes 16,700 lines of Java code and 13,500 lines of C code. Implementation started in February 2003. The first public version, referenced as JUXMEM 0.1 has been released in April 2005.

JuxMem is the central framework based on which a data-sharing service is currently being built, in collaboration with the GRAAL (Lyon) and REGAL (Paris) research groups, within the framework of the GDS (*Grid Data Service*) project of the ACI MD Program (see Section 8.2.2). JuxMem is currently used for transparent data sharing within the following projects: ANR CI LEGO project and ANR MD RESPIRE project. An industrial collaboration with Sun Microsystems has been started in August 2005. JuxMem is also used within several international collaborations: AIST (Tsukuba, Japan), University of Illinois a Urbana Champaign, University of Pisa.

5.6. Dynaco

Keywords: *Grid*, *components*, *framework*, *objects*.

Participants: Françoise André, Jérémy Buisson, Jean-Louis Pazat.

Contact: Jérémy Buisson, `Jeremy.Buisson@irisa.fr`

URL: http://dynaco.gforge.inria.fr/

Status: Version 0.1 is available.

License: GNU Lesser General Public License version 2.1.

Presentation: Dynaco (*Dynamic Adaptation for Components*) is a framework that helps in designing and implementing dynamically adaptable components. This framework is developed by the Paris Project-Team. The implementation of Dynaco is based on the *Fractal Component Model* and its formalism.

In Dynaco, the process of achieving dynamic adaptation is split over three phases:

- Upon the reception of an event that notifies of a change in certain conditions, the component has to make a decision: should it adapt itself to the new situation or not? To do so, it can rely on monitors in order to observe the system. This decision phase is captured by the *Decider Component*.
- Once it has been decided that the component should adapt itself, the component needs to investigate how the adaptation can be achieved. In particular, it has to design the list of the tasks that should be performed. This phase is captured by the *Planner Component*.
- Finally, this adaptation plan has to be executed. The *Executor Component* is the virtual machine that implements the semantics of the instructions used by the Planner Component. To do so, it can rely on the *Modification Controller Components*, which implement some primitive instructions by giving a direct access to the content of the components.

Dynaco mainly defines interfaces between those components. In addition, it includes a reference implementation for the *Julia* implementation of *Fractal*. With this implementation, only the Modification Controller Components are placed in the membrane of the adaptable component.

When the contents of the component encapsulates a parallel code, the Executor Component has to take care of the synchronization between the parallel processes executing the applicative code and the adaptation actions. Our solution for handling this problem relies on a separated framework, called AFPAC.

5.7. Mome

Keywords: *DSM*, *data repository*.

Participant: Yvon Jégou.

Contact: Yvon Jégou, Yvon.Jegou@irisa.fr

Status: Prototype under development.

Contact: Yvon Jégou, http://www.irisa.fr/paris/Mome/welcome.htm

License: APP registration in the future, license type not defined yet (LGPL?).

Presentation: The MOME DSM provides a shared segment space to parallel programs running on distributed memory computers or clusters. Individual processes can freely request mappings between their local address space and MOME segments. The next release of MOME (MOME 1.0) will integrate more dynamicity (adding and removing nodes), better scalability (using a hierarchical implementation), more consistency models (sequential consistency, release consistency, application-managed consistency, parallel reduction), support for background checkpoint consolidation, and better support for the POSIX SMP computation model.

Current status: MOME is implemented in C (50,000 lines). It involved 24 persons-months. The current stable release is MOME 0.8. The next major release MOME 1 will integrate all current developments.

5.8. Vigne

Keywords: *Cluster federation*, *P2P*, *application monitoring*, *grid*, *high availability*, *resource allocation*, *resource discovery*, *transparent data sharing service*.

Participants: Emmanuel Jeanvoine, Christine Morin, Louis Rilling, Thomas Ropars.

Contact: Christine Morin, Christine.Morin@irisa.fr

Status: Under development

License: Not defined yet

Description: VIGNE is a prototype of a grid-aware system for grids, whose goal is to ease the use of computing resources in a grid for executing distributed applications. VIGNE is made up of a set of operating system services based on a peer-to-peer infrastructure. This infrastructure currently implements a structured overlay network inspired from *Pastry* [95] and an unstructured overlay network inspired from *Scamp* [79] for join operations. On top of the structured overlay network, a transparent data sharing service based on the sequential consistency model has been implemented. It is able to handle an arbitrary number of simultaneous reconfigurations. An application execution management service has also been implemented including resource discovery, resource allocation and application monitoring services.

The VIGNE prototype has been developed in C and includes 39,000 lines of code. This prototype has been coupled with a discrete-event simulator. The use of this simulator enabled to evaluate the VIGNE system in systems composed of a large number of nodes. In 2006, the VIGNE system has been experimented on several sites of the GRID 5000 platform and in the framework of the *SALOME* open source integration platform for numerical simulation (http://www.salome-platform.org/) at EDF R&D.

6. New Results

6.1. Introduction

Research results are presented according to the scientific challenges of the PARIS Project-Team. This year we added a new section that describes how our research activities are integrated within the *CoreGRID Network of Excellence*, in which PARIS is actively involved.

6.2. Operating system and runtime for clusters and grids

Keywords: *Cluster*, *MPI*, *checkpointing*, *cluster federation*, *data stream migration*, *distributed shared memory*, *distributed system*, *distributed file system*, *fault tolerance*, *global scheduling*, *grid*, *high availability*, *high performance communication*, *operating system*, *peer-to-peer*, *process migration*, *resource management*, *self-healing system*, *self-organizing system*, *single system image*.

6.2.1. Kerrighed

Participants: Matthieu Fertré, Pascal Gallard, Adrien Lèbre, Renaud Lottiaux, Christine Morin, Jean Parpaillon.

The PARIS Project-Team is engaged in the design and development of KERRIGHED, a genuine *Single System Image* (SSI) cluster operating system for general-purpose, high-performance computing [91]. A genuine SSI offers users and programmers the illusion that a cluster is a single high-performance and highly-available computer, instead of a set of independent machines interconnected by a network. A SSI should offer four properties:

Resource distribution transparency: Offering processes transparent access to all resources, and resource sharing between processes whatever the resource and process location.

High performance.

High availability: Tolerating node failures and allowing application checkpoint and restart.

Scalability: Dynamic system reconfiguration, node addition and eviction, transparently to applications.

6.2.1.1. Kerrighed evolutions.

In 2006, a major refactoring of KERRIGHED has been carried out. It consists in porting the previous stable version of the system based on Linux 2.4 kernel, to Linux 2.6.11 kernel.

A port of KERRIGHED on *User Mode Linux* (UML) architecture of virtual machine has also been done in 2006. The UML version of KERRIGHED is useful to facilitate the debugging of the system, and for demonstration purposes.

The robustness of KERRIGHED has been significantly enhanced, and several new functionalities have been implemented such as high-availability mechanisms to automatically reconfigure KERRIGHED services in the event of the addition or eviction of a hot node [59]. KERRIGHED V2.0 version has been released at the end of the COCA contract in March 2006 [60], [58], [61].

In 2006, we have evaluated the potential of KERRIGHED for various application domains, different from the scientific applications: bio-informatics (internship of Jérôme Gallard [57]) and Web services (internship of Robert Guziolowski [62]).

A start-up, the KERLABS SARL Company (http://www.kerlabs.com/), has been created in October 2006 by Pascal Gallard, Renaud Lottiaux and Louis Rilling in order to transfer the KERRIGHED technology. KERLABS has been hosted by the INRIA *Emergys* incubator since February 2006. KERLABS will continue the development and the industrialization of the KERRIGHED technology, to deliver systems specifically suited to the management of clusters. KERLABS intends to promote and develop a community of users and developers around the original KERRIGHED free software.

6.2.1.2. SSI-OSCAR package.

OSCAR (http://oscar.openclustergroup.org/) is a distribution for Linux clusters which provides a snapshot of the best known methods for building, programming and using clusters. We have worked with the OSCAR Team at the Oak Ridge National Laboratory, Oak Ridge, Tennessee, USA, in order to maintain the SSI-OSCAR package which integrates KERRIGHED in the OSCAR software suite. By combining OSCAR with the KERRIGHED Single System Image operating system, a cluster becomes easy to install, administrate, use and program.

Since 2005, the SSI-OSCAR package has become a standard third-party OSCAR package. As such, SSI-OSCAR is automatically proposed as all other official OSCAR packages (only core packages are directly included in the OSCAR suite, third-party packages being available through on-line repositories). In 2006, we have implemented a tool to automatically generate RPM and Debian packages as well as the corresponding SSI-OSCAR packages from KERRIGHED source code. We have revisited the building process of KERRIGHED to conform to the *de facto* building standards used in the Linux community. We have also participated in the design of OSCAR 5.0, which implements a more modular architecture than the previous OSCAR versions. SSI-OSCAR packages have been implemented to conform to this new OSCAR version.

6.2.1.3. Tool for testing Kerrighed.

In the framework of the ODL contract of Jean Parpaillon (*Opération de développement logiciel*, INRIA short-term software development initiative), we have implemented a software testing tool. The goal of this tool is to automatically test KERRIGHED software and the SSI-OSCAR packages. The tool automatically builds a KERRIGHED binary image from the source code of the development version (available at the INRIA Gforge repository). Test results are made available to developers through a DART server. Thanks to this tool, the compilation process of KERRIGHED can now be automatically tested on each night. This tool allows detecting the regressions introduced during the development of KERRIGHED. We plan to extend this tool to perform execution tests as well, for instance to validate the conformance of the KERRIGHED system to the POSIX standard, using traditional test suites provided by the Linux community. The main issue is to deal with failing tests, leading to a crash of KERRIGHED system.

6.2.1.4. Distributed file system.

KerFS, a KERRIGHED distributed file system, has been designed and implemented to exploit the disks attached to cluster nodes. *KerFS* provides a unique, cluster-wide naming space, and enables to store the files of a directory in multiple disks throughout the cluster. It has been implemented based on the *container* concept, originally proposed for global memory management. Containers are used to manage not only memory pages, but also generic objects. The meta-data structures of the file system are kept consistent cluster-wide using object containers.

In 2006, in the framework of the XTREEMOS project, we have started to revisit the design of *KerFS* to improve its performance. We have worked on providing parallel accesses to files split and/or replicated on the cluster disks. Also, we have designed an I/O scheduler to improve the behavior of the disk I/O system when the cluster nodes are used for executing multiple concurrent applications. We are now targeting the implementation of the proposed mechanisms. We will also study how fault-tolerance mechanisms can be integrated in *KerFS*.

6.2.1.5. Application Management.

One of the advantages of SSI operating system for clusters is that users can launch applications interactively, in the same way as they do when using a single PC running Linux. However, when multiple users launch applications on the same cluster (typically when a cluster is used as a departmental server), it may happen that the workload exceeds the cluster capacity. To avoid this situation, a solution is to execute a batch system on top of the SSI operating system. However, this makes the submission of applications more difficult for users who need to provide a description of their applications.

We have investigated a different approach, which consists in integrating a *fork-delay* mechanism in a SSI cluster operating system to delay the execution of processes when the cluster is overloaded (Jérôme Gallard internship [57]). When an application is launched with the fork-delay capability enabled, its processes are queued if the cluster is overloaded. When a process terminates its execution, the global scheduler resumes the execution of the delayed processes, if any. At any time, if the cluster load is too high, the global scheduler may decide to suspend the execution of some processes. We have validated this approach with the implementation of a first prototype in KERRIGHED. We plan to refine the distributed management of the delayed process queue, to take into account dependencies between processes belonging to the same application (*gang scheduling*).

6.2.1.6. High Availability.

KERRIGHED is a distributed system made up of co-operating kernels executing on the cluster nodes.Therefore, a node failure has a significant impact on the operating system itself, not only on the applications being executed on top of the system. We have implemented a generic service to be used by the various services composing the KERRIGHED operating system to enable their automatic reconfiguration when a node is added or removed in the cluster [59].

6.2.1.7. Monitoring.

Monitoring is of uttermost importance to achieve robust computing. Monitoring is needed for failure and attack detection. It can also be used for system management and load balancing. A monitoring system must enjoy several properties. It should be non-intrusive (no need to modify the target OS), tamper-proof (no possible intrusion), and autonomous (no involvement of the target OS). It should provide a consistent view of the distributed OS state, and be customizable for flexibility. Moreover, it should be based on fail-safe communications.

In the context of the PHENIX Associated Team, we have investigated the design of a monitoring system for KERRIGHED based on the backdoor architecture developed in the DiscoLab laboratory of Rutgers University (Benoît Boissinot internship [56]). Autonomy is achieved thanks to a monitoring system based on a virtualization technology.

Our proposition is a *distributed virtual backdoor architecture*. The idea to monitor the operating system running on a PC is to execute the backdoor and the monitored OS in different virtual machines on top of a virtual machine monitor. The main issue to be tackled in the implementation is the extraction of OS state from the memory. As KERRIGHED is a distributed system running on multiple machines, a co-operation protocol has been designed to compute a consistent global state from the partial information gathered by each virtual backdoor. We have implemented the proposed architecture on top of the *Xen* virtual machine monitor to demonstrate how distributed virtual backdoors can co-operate to monitor a distributed state.

6.2.2. Grid-aware Operating System

Participants: Emmanuel Jeanvoine, Yvon Jégou, Adrien Lèbre, Sandrine L'Hermitte, David Margery, Christine Morin, Louis Rilling, Thomas Ropars, Oscar Sanchez.

6.2.2.1. The Vigne System.

Our research aims at easing the execution of distributed computing applications on computational grids. These grids are composed of a large number of geographically-distributed computing resources. This large-scale distribution makes the system dynamic: failures of single resources are frequent (interconnecting network failures, and machine failures), and any participating entity may decide at any time to add or remove nodes from the grid.

To ease the use of such dynamic, distributed systems, we propose to build a distributed operating system which provides a Single System Image, which is self-healing, and which can be tailored to the needs of the users [42]. Such an operating system is composed of a set of distributed services, each of them providing a Single System Image for a specific type of resource, in a fault-tolerant way. We are implementing this system on a research prototype called VIGNE [43]. Experimental evaluations are made on the GRID 5000 research grid. The work of Year 2006 is twofold.

First, we have mainly worked on four services of the VIGNE system that are: resource discovery, resource allocation, monitoring and system interface [36], [37], [50]. We have extended the resource discovery service by designing and implementing a new resource discovery protocol called *Random Walk Optimized for Grid Scheduling* (RW-OGS). RW-OGS uses learning and broadcasting strategies to improve the quality of the results obtained after a resource discovery. Thus, the resource discovery service provides less loaded resources to the resource allocation service [35], and the efficiency of the global resource allocation is increased.

The resource allocation service has been extended to handle co-scheduled tasks. VIGNE provides system features for coordinating tasks before execution, and for enriching the environment of each task with additional information useful for co-scheduling (real location of co-scheduled-tasks and machine file). Thus, VIGNE is able to execute complex applications, like MPI or master/worker applications. A monitoring service has been designed and implemented with the aim to provide grid users with fine-grained information about the application execution (internship of Thomas Ropars [64], [53]). This service is designed for a large-scale grid since it consumes very little bandwidth. To monitor applications, it uses a pre-loaded dynamic library that overloads some system calls like `fork`, `waitpid` or `exit`. Thus, it is able to detect crashes of the application processes, what is a keystone for reliable application execution and fault-tolerance policies. The service also provides accurate information about resource consumption. The system interface has been extended to allow job submission to the *OpenPBS* batch-scheduler through VIGNE. Thus VIGNE now handles three kinds of resources: Linux workstations, KERRIGHED clusters, and OpenPBS clusters.

Second, we have worked on an integration between VIGNE and the SALOME platform for numerical simulation (http://www.salome-platform.org/). We have designed and implemented a plug-in in the SALOME platform in order to allow the SALOME applications to be executed through VIGNE. The plug-in wraps SALOME instructions for job submission into VIGNE queries, and it automatically deploys the linked libraries of the SALOME applications. We have also extended VIGNE to provide system features for handling input/output files, and for extending jobs from the SALOME platform.

6.2.2.2. The XtreemOS Linux-based Grid Operating System.

The European Integrated Project XTREEMOS [65], coordinated by Christine Morin, addresses Section 2.5.4 of the 2006 Work Programme: *Advanced Grid Technologies, Systems and Services*. It was launched in June 2006. The overall objective of the XTREEMOS Project is the design, implementation, evaluation and distribution of an open source Grid operating system, with a native support for virtual organizations (VO). The proposed approach is the construction of a Grid-aware OS made up of a set of system services based on the traditional general-purpose Linux OS, extended as needed to support VO and to provide appropriate interfaces to the Grid OS services. The XTREEMOS consortium includes 19 academic and industrial partners. Various end-users are involved in the XTREEMOS Consortium, providing a wide range of test cases in scientific and business computing domains.

In 2006, apart from setting up the project, we have worked on the specification of XTREEMOS operating system along three main directions.

6.2.2.3. XtreemOS flavor for clusters.

We have specified the cluster flavor of XTREEMOS, *LinuxSSI*, which leverages KERRIGHED technology. We plan to develop an efficient cluster file system and to provide mechanisms to tolerate reconfiguration events in a scalable way [67].

6.2.2.4. Checkpointing Service.

We have started to design a modular Grid checkpointer architecture [67], [66]. The proposed architecture is hierarchical, involving a grid-level checkpointer, a system-level scheduler, and a kernel-level checkpointer. The grid checkpointer is in charge of coordinating the checkpointing protocols for applications composed of several units executed on multiple grid nodes. The system checkpointer is in charge of checkpointing an application unit on a single grid node. The kernel checkpointer, triggered by the system checkpointer, extracts, saves and restores the state of a process or thread on a grid node.

In the standard flavor of XTREEMOS for individual PCs, the kernel checkpointer will be based on *BLCR*, which is one of the most advanced open-source implementation of a checkpoint/restart system for Linux. We plan to augment BLCR with the following features:

- Save the shared libraries used by the process in the checkpoint, rather than assume that they will be present on the system when the process will be restarted.
- Save the security context (VO specific information) in the snapshot of a process.

- Extend saving of the snapshot from a specific file to a generic file descriptor, so that checkpoints can be stored in a grid object in the future.
- At restart, provide information to the restarted process about the changes in the environment (process id, IP address, host name).

We will also study checkpointing strategies to be implemented in the system and grid checkpointers for large-scale applications executed on top of XTREEMOS.

6.2.2.5. Virtual Organization Management

We have specified the overall approach for Virtual Organization management in XTREEMOS [66]. The management work of a VO involves two levels: the VO level (or global level) and the node level (to be implemented as extensions to the Linux operating running on each grid node). The VO-level management includes membership management of users and nodes that join in or leave from a VO, policy management (e.g., group and role assignment), and runtime information management (e.g., querying active processes or jobs in a VO). The main responsibilities of node-level management include: translating from grid identities into local identities; granting or denying access to resources (files, services, etc.); checking limitations of resource usage (CPU wall time, disk quotas, memory, etc.); protecting and separating of resource usage by different users; logging and auditing of resource usage, etc.

XTREEMOS supports VO management by the co-operative activities of VO-level and node-level management services. The key challenge here is to co-ordinate VO-level policies and local policies on nodes which depend on autonomic domain administrators. On the one hand, the enforcement of multiple VO security policies should be differentiated, while on the other hand, this kind of enforcement should not be conflicting with any local policy of nodes and it should not impair the usability of resources for grid users.

6.3. Middleware for computational grids

6.3.1. Parallel CORBA objects and components

Keywords: *CORBA*, *Grid*, *distributed component*, *distributed object*, *parallelism*.

Participants: Hinde Lilia Bouziane, Raúl López Lozano, Christian Pérez, Thierry Priol.

The concept of (distributed) parallel object/component appears to be a key technology for programming (distributed) numerical simulation systems. It joins the well-known object/component oriented model with a parallel execution model. Hence, a data distributed across a parallel object/component can be sent and/or received almost like a regular piece of data while taking advantage of (possible) multiple communication flows between the parallel sender and receiver.

The PARIS Project-Team has been working on such a topic for several years. PACO was the first attempt to extend CORBA with parallelism. PACO++ is a second attempt that supersedes PACO in several aspects. It is a generic extension to CORBA, so that it can be added to any implementation of CORBA. It considers that the parallelism of an object is mainly an implementation issue: it should not be visible to users, but in some specific occasions. Hence, the OMG IDL is no longer modified. GRIDCCM is the evolution of PACO++ into the component model of CORBA.

The work carried out in 2006 was related to the improvement of PACO++ and with the beginning of the implementation of GRIDCCM, a parallel CORBA component model.

Future work will mainly concern the development of GRIDCCM and its validation within two ANR projects. The support of PACO++ will be continued, as it is a foundation of GRIDCCM.

6.3.2. Dynamic software component models

Keywords: *CORBA Component Model (CCM)*, *Grid*, *dynamic behavior*, *software component*.

Participants: Hinde Lilia Bouziane, Christian Pérez, Thierry Priol.

Software component models are succeeding in handling another level of the software complexity by dealing with its architecture. However, a current limitation is that only *static* architectures can be handled with. Dynamic behaviors, like the well-known master-worker pattern, are not expressible. It is an important lack as many applications require such a feature.

Our objective is to study how to capture such dynamic behavior within a component model.

We have proposed to use the concept of *abstract collection* and of *request delivery policy pattern* to support the master-worker design pattern in component-based application. From an application point of view, the master-worker behavior is well captured. The model is valid for ADL-based models like CCM and *Fractal*, as well as for models without ADL like *CCA*. We have achieved a first integration of the *DIET* middleware system as an advanced request delivery policy. Two sequential applications coming from two INRIA projects (*VISAGE* and *VISTA*) have been parallelized with respect to our model. We have merged our master-worker model with our data-sharing model between components.

Two extensions of this work are considered. First, we will perform benchmarking of the model with respect to synthetic benchmarks and real applications. Second, we plan to use the DYNACO adaptive framework to deal with the management of a dynamic set of workers.

6.3.3. Application deployment on computational grids

Participants: Landry Breuil, Boris Daix, Christine Morin, Thierry Priol.

The deployment of parallel component-based applications is a critical issue in the utilization of computational Grids. It consists in selecting a number of nodes and in launching the application on them. We proposed a generic deployment model that aims to automatically deploy complex applications on grids. The core of the model is a *Generic Application Description model* (GADe) that enables to decouple most of the deployment tool from a specific application description. We have also proposed a description model for grid networks that provides a *synthetic* view of the network topology.

Year 2006 was mainly devoted to the stabilization of the prototype named ADAGE. A new feature of redeployment has been added. The outputs of the deployment of an application can been used in the future deployment operation either to replay the same deployment, or to add elements to a running application. For example, it enables to add more workers to a running component-based, master-worker application. We have also started to study the support of PADICOTM.

Our current deployment model is restricted to static applications. Hence, we have started to revisite the model to support dynamic applications.

The next major goal is to proposed a model for dynamic applications. We also plan to finish the support of PADICOTM.

6.3.4. Adaptive components

Keywords: *Grid*, *components*, *framework*, *objects*.

Participants: Françoise André, Jérémy Buisson, Jean-Louis Pazat.

Since grid architectures are also highly dynamic, using resources efficiently on such architectures is a challenging problem. Software must be able to dynamically react to the changes of the underlying execution environment. In order to help developers to create reactive software for the grid, we are investigating a model for the adaptation of parallel components.

We have defined a parallel, self-adaptable component as a parallel component which is able to change its behavior according to the changes of the environment. Based on our model for the adaptation of parallel component, on our previous experience, and on a collaboration with the University of Pisa [1], we have defined a generic model of dynamic adaptation and used it to define frameworks.

We have separated the framework in two parts: a generic framework for dynamic adaptation of components, and a specific implementation for synchronizing parallel SPMD codes for adaptations.

The generic framework (DYNACO) implements the generic adaptation framework. It is a *Fractal* component-based implementation.

The specific parallel synchronization tool has been redesigned from the previous implementation to separate the parallel specific part from the generic framework.

The DYNACO framework has been used in the *COA* INRIA Cooperative Research Action (ARC) to demonstrate its use for computational steering of a simulation application.

In the next future, we will study the impact of dynamic adaptation on the design of resource allocators and batch schedulers. Use of adaptive components will also be studied for dependable grid computing within the context of the *SafeScale* Project.

6.3.5. Dynamic Load Balancing

Keywords: *Grid*, *load balancing*.

Participants: Jean-Louis Pazat, Nagib Abi Fadel.

Dynamic load balancing algorithms have proven to be better than static load balancing algorithms. However, in many applications one algorithm cannot be the best one during the whole life of the application, especially in multi-phase applications. We are studying the dynamic adaptation of load-balancing algorithms.

We first studied the basic characteristics of dynamic load balancing algorithms on Grids. We divided algorithms into two categories: *centralized* and *totally distributed* algorithms. We also studied some evaluation function to be able to choose at runtime between two algorithms.

We have implemented a prototype for changing dynamically a load balancing algorithm using the *AMPI* software and the *Charm++* library which includes some load balancing algorithms.

In the next future, we will study the integration within the DYNACO framework

6.4. Large-scale data management for grids

6.4.1. The JuxMem data-sharing service

Keywords: *DSM*, *JXTA*, *Peer-to-peer*, *grid data sharing*.

Participants: Gabriel Antoniu, Luc Bougé, Loïc Cudennec, Mathieu Jan, Sébastien Monnet.

Since 2003, we have been working on the concept of *data-sharing service* for grid computing, that we defined as a compromise between two rather different kinds of data sharing systems: (1) *DSM systems*, which propose consistency models and protocols for efficient transparent management of *mutable data, on static, small-scaled configurations (tens of nodes)*; (2) *P2P systems*, which have proven adequate for the management of *immutable data* on *highly dynamic, large-scale configurations (millions of nodes)*. We illustrated this concept through the JUXMEM software platform. The main challenge in this context is to define appropriate models and protocols allowing to guarantee both *consistency* of replicated data and *fault tolerance* in *large-scale, dynamic environments*.

To tackle the issues described above, we have defined an architecture proposal for a data sharing service. This architecture mirrors a federation of distributed clusters and is therefore *hierarchical* and is illustrated through a software platform called JUXMEM (for *Juxtaposed Memory*). Its architecture consists of a network of peer groups (`cluster` groups), each of which generally corresponds to a cluster at the physical level. All the groups are inside a wider group which includes all the peers which run the service (the `juxmem` group). Each `cluster` group consists of a set of nodes which provide memory for data storage (called *providers*). All providers which host copies of the same data block make up a `data` group, to which is associated an ID. To read/write a data block, clients only need to specify this ID: the platform transparently locates the corresponding data block. This architecture is illustrated by a software prototype (development started in February 2003, currently in progress). The prototype is based on the JXTA [69] generic peer-to-peer framework, which provides basic building blocks for user-defined peer-to-peer services.

In 2006, we have worked on the integration of the transparent data access model provided by JuxMem, into different grid programming models. We proposed an approach integrating data sharing with component-based programming environments [27], [26]. Within the GDS Project of the ACI MD program, we have explored integrating data sharing into GridRPC, component-based programming environments. To validate this latter approach, JuxMem has been integrated into the *DIET* GridRPC environment developed within by the *GRAAL* Project-Team of Inria Rhône-Alpes. Several large-scale experiments have been completed on the Grid 5000 testbed [45]. A paper describing the results has been recently been submitted to an international conference. A common data-sharing architecture allowing JuxMem to be coupled with the *ASSIST* environment developed at the University of Pisa has also been proposed (work completed within the CoreGRID NoE, WP3) and published at the CoreGRID Integration Workshop [25].

6.4.2. Large-scale evaluation of JXTA protocols on grids

Keywords: *JXTA*, *peer-to-peer*, *performance evaluation*.

Participants: Gabriel Antoniu, Mathieu Jan, Loïc Cudennec.

Features of the P2P model, such as scalability and volatility tolerance, have motivated its use in distributed systems. Several generic P2P libraries have been proposed for building distributed applications. However, very few experimental evaluations of these frameworks have been conducted, especially at large scales. Such experimental analyzes are important, since they can help system designers to optimize P2P protocols, and better understand the benefits of the P2P model. This is particularly important when the P2P model is applied to special use cases, such as grid data sharing. In collaboration with Sun Microsystems, we have evaluated the scalability of two main protocols proposed by the JXTA P2P platform: the rendezvous protocol, whose role is to set up and maintain the JXTA P2P overlay, and the discovery protocol, used to find resources inside a JXTA network. We performed a detailed, large-scale, multi-site experimental evaluation of these protocols, using up to 580 nodes spread over the nine clusters of the French Grid'5000 testbed. Mathieu Jan presented the results at the Grid'5000 school (Grenoble, March 2006) and obtained the Best Presentation Award. A paper has been submitted to an international conference.

6.4.3. Dynamic deployment of grid services and applications

Keywords: *dynamic deployment*, *grids*.

Participants: Gabriel Antoniu, Luc Bougé, Loïc Cudennec.

To experiment distributed services on large-scale testbeds, dedicated deployment tools are used, such as ADAGE (developed within the Paris Project-Team). Currently, such tools take as an input the application description (in terms of logical participating entities) and the resource description (in terms of nodes and network) and define a valid and possibly optimized mapping between the two. However, deployment is static, in the sense that the set of application entities and the set of resources are fixed at deployment time. We addressed this limitation by defining a mechanism allowing services with dynamic needs to interact with the grid resource manager and dynamically allocate the needed resources. To validate our approach, we implemented a prototype for the JuxMem data-sharing service. This work has been carried out during Voichita Almasan's Master internship, co-advised by Luc Bougé, Gabriel Antoniu and Loïc Cudennec.

6.4.4. Adaptive application-driven overlays

Keywords: *data sharing*, *fault-tolerance*, *group membership*, *peer-to-peer*.

Participants: Gabriel Antoniu, Sébastien Monnet.

In collaboration with Indranil Gupta's team from the University of Illinois at Urbana Champaign, we studied techniques to use probabilistic fault-tolerance strategies to support data consistency in distributed applications based on dynamic collaborative groups. We proposed, implemented and evaluated an application-driven peer-to-peer overlay, which adapts its logical topology to the application usage. The first results have been published at SRDS 2006 [52] and an extended paper has been submitted to a journal.

6.4.5. *The Mome data-repository*

Participant: Yvon Jégou.

Providing the data to the applications is a major issue in grid computing. The execution of an application on some site is possible only when the data of the application are present on the "data-space" of this site. It is thus necessary to move the data from the production sites to the execution sites. Using a *Distributed Shared Memory* for sharing data objects on the Grid has been shown to facilitate the execution of applications on the Grid. However, traditional DSM systems have been developed for clusters of computers and target simple applications. Grid systems are much more dynamic (nodes can be dynamically added or removed) and more heterogeneous (at least in the interconnection layer). Grid applications are more complex.

The recent developments on the MOME DSM allow to dynamically manage the DSM nodes (add and remove), to consider the Grid interconnection structure through hierarchical management of the nodes, and to dynamically manage the shared space of the applications using a new memory allocator.

6.5. Advanced computation models for the Grid

This work is carried out in close co-operation with Pascal Fradet, from INRIA Rhône-Alpes (Project-Team *POP ART*).

We are considering unconventional approaches for Grid programming and, more generally, for the programming of distributed applications.

It is well known that the task of programming is very difficult in general and even harder when the environment is distributed. As usual, the best way to proceed is by separation of concerns. Programs are first expressed in a model independent of any architecture, and then are refined taking into account the properties of the (distributed) environment. Several properties have to be taken into account, such as correctness, coordination/co-operation, mobility, load balancing, migration, efficiency, security, robustness, time, reliability, availability, computing/communication ratio, etc.

Our present work relies on the chemical reaction paradigm and more precisely on the GAMMA model of programming. We believe that this model can be a nice basis for the construction of applications exploiting grid technology.

Our recent contributions include the extension of GAMMA to higher-order and the generalization of multiplicity. The extension of the basic GAMMA model to a higher-order GAMMA makes it possible to consider a GAMMA program as a member of a multiset, thus eligible for reactions as any other element of the multiset. We have called this model, the γ-calculus.

The *Grand Challenge in Non-Classical Computation* Workshop has been a great opportunity to expose our model, and to have a large overview on non-conventional models of computation. It has also raised some fundamental questions about non-classical programming languages. Revised versions [19], [22] of the two related articles will be published by the end of this year in the *International Journal of Unconventional Computing*.

Another generalization of the GAMMA language stands in the introduction of multisets with infinite cardinality and multisets with a negative cardinality. These new kind of data structures, combined with the above higher-order properties, provide a very general and powerful tool for expressing very general (and original) coordination schemes. This work has been presented in the *International Workshop on Developments in Computational Models* which has been published in a volume [30] of the Electronic Notes in Theoretical Computer Science (ENTCS) series. A complete version of this work has been published in a journal this year [18].

We have also published an abstract [54] in a special issue of *Ercim News* devoted to "Emergent Computing".

Our most recent work concerns task coordination on Grids within the chemical framework. In a first step, applications are programmed in an abstract manner describing essentially the chemical coordination between (not necessarily chemical) software components. In a second step, chemical service programs are specifically provided to the run-time system in order to obtain the expected quality of service from the resources, in terms of efficiency, reliability, security, etc. The implementation of a prototype is being continued.

6.6. Experimental Grid Infrastructure

Participants: Yvon Jégou, David Margery, Pascal Morillon.

The PARIS Project-Team manages an experimental computation platform dedicated to operating system, runtimes, middleware, grid and P2P research. This platform is now integrated in the nation-wide grid infrastructure GRID 5000. At the end of 2005, our platform was made up of 66 dual Intel Xeon from Dell (January 2004), 33 dual Xserve G5 from Apple (September 2004), 66 dual AMD Opteron from Sun (October 2005), and 100 dual AMD Opteron from HP (November 2005). Our platform contains now more than the targeted 500 processors.

The interconnection of our platform to the other GRID 5000 platforms has been been upgraded during 2006 to use the Renater 4 infrastructure, enabling a 10 Gb/s interconnection to the other GRID 5000 sites. At this occasion, the topology of the local network has been simplified, each node being connected to the other ones through a non-blocking switch.

In 2006, we have also proceeded to initial tests to interconnect our experimental infrastructure to another experimental platform, DAS-3, located in The Netherlands. These tests are meant to prepare interconnection of all of the GRID 5000 sites to DAS-3 in 2007.

Year 2007 will also see the arrival of the first multi-core machines in our experimental platform. Our platform will also be used as a testbed for experiments originating from our partners of XTREEMOS and in the COREGRID Network o Excellence.

7. Contracts and Grants with Industry

7.1. EDF 1

Participants: Christine Morin, Emmanuel Jeanvoine.

Program: The collaboration with EDF R&D aims at designing, implementing and evaluating a resource discovery and allocation service for a cluster federation.

Starting time: October 1, 2004

Ending time: September 30, 2007

Partners: EDF R&D, INRIA

Support: EDF R&D funding, PhD CIFRE grant (Emmanuel Jeanvoine)

Project contribution: The work carried out by the PARIS Project-Team relates to the design and implementation of a Grid-aware operating system for cluster federations. As part of this contract, we design a resource discovery and allocation service based on an underlying, peer-to-peer overlay network. It enables to cope with the decentralized and dynamic nature of a cluster federation. We also study application scheduling policies for cluster federations that will be evaluated experimentally with workloads provided by EDF R&D.

7.2. EDF 2

Participants: Boris Daix, Christine Morin, Christian Pérez.

Program: The collaboration with EDF R&D aims at designing, implementing and evaluating a resource discovery and allocation service for a cluster federation.

Starting time: January 1, 2006

Ending time: December 31, 2008

Partners: EDF R&D, INRIA

Support: EDF R&D funding, PhD CIFRE grant (Boris Daix)

Project contribution: The work carried out by the PARIS Project-Team relates to the dynamic deployment of coupled, parallel scientific applications on federations of clusters taking into account their execution constraints.

7.3. Sun Microsystems

Participants: Gabriel Antoniu, Luc Bougé, Loïc Cudennec, Mathieu Jan, Thierry Priol.

Starting time: October, 2005

Ending time: September, 2008

Partners: Sun Microsystems,INRIA

Support: Sun funding, PhD grant (*Loïc Cudennec*)

Project contribution: The work studies techniques to optimize the use of the JXTA P2P library on grid infrastructures. In 2006 we performed scalability studies for the JXTA rendezvous protocol (which is the central protocol used to build the JXTA overlay). Mathieu Jan presented the results at the Grid'5000 school (Grenoble, March 2006) and obtained the Best Presentation Award. A second goal of this collaboration is to adequately support the interaction between JXTA-based applications and grid resource management systems, in order to cope with dynamic resources. Work is in progress on this topic.

8. Other Grants and Activities

8.1. Regional grants

PhD grants: The Brittany Regional Council provides half of the financial support for the PhD theses of Mathieu Jan (starting on October 1, 2003, for 3 years) and Loïc Cudennec (starting on October 1, 2005, for 3 years). This support amounts to a total of 28,000 Euros/year.

8.2. National grants

8.2.1. ANR CI: ANR Program on High Performance Computing and Simulation

8.2.1.1. ANR CI DISC

Participants: Raúl López Lozano, Christian Pérez, Thierry Priol.

The *DISC* Project of the ANR CI gathers 7 partners: 6 academic research teams – the *CAIMAN*, *SMASH* and *OASIS* Project-Teams from INRIA Sophia-Antipolis, the PARIS Project-Team from IRISA, the *MOAIS* Project-Team from INRIA Rhône-Alpes and Laboratory ID-IMAG, and the *Distributed Systems and Objects* Team from LaBRI, and one industrial partner – EADS CCR.

It aims at studying and promoting a new paradigm for programming non-embarrassingly parallel scientific computing applications on distributed, heterogeneous, computing platforms. The *DISC* Project concentrates its activities on numerical kernels and related issues that are of interest to a large variety of application contexts. The emphasis is put on designing parallel numerical algorithms and programming simulation software that efficiently exploit a computational grid and more particularly, the GRID 5000 testbed.

It is a 3-year project which started in January 2006. Project site: http://www-sop.inria.fr/caiman/personnel/Stephane.Lanteri/discogr

8.2.1.2. ANR CI LEGO

Participants: Gabriel Antoniu, Loïc Cudennec, Hinde Lilia Bouziane, Christian Pérez.

The *LEGO* Project of the ANR CI gathers 6 partners: LIP – INRIA Project-Team *GRAAL*; IRISA– INRIA Project-Team PARIS; LaBRI – INRIA Project-Team *Runtime*; the IRIT Laboratory in Toulouse; and the *CRAL* Center of Astronomical Research of Lyon.

The aim of this project is to provide algorithmic and software solutions for large-scale architectures; the focus is on performance issues. The software component approach provides a flexible programming model where resource management issues and performance optimizations are handled by the implementation. On the other hand, current component technology does not provide adequate data-management facilities, needed for large data in widely distributed platforms, and does not deal efficiently with dynamic behaviors. The project addresses topics in programming models, communication models, and scheduling. The results are validated on three applications: an ocean-atmosphere numerical simulation, a cosmology simulation, and a sparse matrix solver.

It is a 3-year project which started in January 2006. Project site: http://graal.ens-lyon.fr/LEGO/.

8.2.1.3. ANR CI NUMASIS

Participant: Christian Pérez.

The NUMASIS Project of the ANR CI gathers 8 partners: two industrial companies – BULL (Echirolles) and Total (Pau), two EPIC institutions – BRGM (Orléans) and CEA (Bruyères-le-Châtel), and 4 academic laboratories – ID-IMAG (INRIA Project-Teams *Mescal* and *Moais*), LaBRI (INRIA projects-Teams *Runtime* and *Scalapplix*), LMA (INRIA Project-Team *Magique 3D*) and IRISA (INRIA Project-Team PARIS).

It deals with recent NUMA multiprocessor machines with a deep hierarchy. In order to efficiently exploit it, the project aims at evaluating the features of current systems, at proposing and implementing new mechanisms for process, data and communication management. The target applications come from the seismology field that appear representative of current needs in scientific computing.

It is a 3-year project which started in January 2006. Project site: http://numasis.gforge.inria.fr/.

8.2.2. ANR MD: ANR Program on Data Masses and Ambient Knowledge

The PARIS Project-Team is involved in the ACI MD (MD for *Masses de Données*) Program of the Ministry of Research, now continued as the ANR MD Program of the newly-founded *National Research Agency*. It aims at fostering research activities in the area of large-scale data management, including Grid computing. The first call for proposal was issued in 2003. The following paragraphs give a short overview of the project-team involvement in this initiative.

8.2.2.1. ANR MD GDS

Participants: Gabriel Antoniu, Luc Bougé, Loïc Cudennec, Mathieu Jan, Sébastien Monnet.

The GDS Project of the ANR MD (formerly ACI MD) gathers 3 research teams: PARIS (IRISA), *REGAL* (LIP6) and *ReMaP/GRAAL* (LIP). The main goal of this project is to specify, design, implement and evaluate a data sharing service for mutable data and integrate it into the *DIET* ASP environment developed by *ReMaP/GRAAL*. This service is built using the generic JUXMEM platform for peer-to-peer data management (currently under development within the PARIS Project-Team, see section 6.4.1). JUXMEM can implement and compare multiple replication and data consistency strategies defined together by the PARIS and *REGAL* research groups. The project started in September 2003 and ended in September 2006. It was coordinated by Gabriel Antoniu (PARIS). Project site: http://www.irisa.fr/GDS/.

Three PhD students have been working within this project in 2006. M. Jan defended his PhD on November 20, 2006. It was devoted to the general design of the JUXMEM platform. S. Monnet defended his PhD on November 30, 2006. It was devoted to implementing consistency/fault-resiliency protocols in JUXMEM. The last one, L. Cudennec, joined the project in October 2005. He works on the deployment of JUXMEM on large, distributed federations of clusters, whose a typical instance is the GRID 5000 experimental platform.

8.2.2.2. ANR MD GdX

Participants: Gabriel Antoniu, Luc Bougé, Loïc Cudennec, Mathieu Jan, Sébastien Monnet, Thierry Priol.

The *Data Grid Explorer* (GdX) Project aims at implementing a large-scale emulation tool for the communities of a) distributed operating systems, b) networks, and c) the users of Grid or P2P systems. This large-scale emulator (http://www.lri.fr/~fci/GdX/) consists of a database of experimental conditions, a large cluster of 1000 PCs, and tools to control and analyze experiments. The project includes studies concerning the instrument itself, and others that make use of the instrument. The GDS Project of the ANR MD Program (formerly ACI MD), coordinated by the PARIS Project-Team, is a partner of GdX, as a *user project*. The project started in September 2003 and ended in September 2006. Gabriel Antoniu is the local correspondent of *GdX* for the PARIS Project-Team. In 2006, large-scale experiments were performed by Mathieu Jan on the GdX platform, with the goal of evaluating the scalability of the JXTA P2P platform (used as underlying support for the data sharing service developed within the GDS project).

8.2.2.3. ANR MD RESPIRE

Participants: Gabriel Antoniu, Luc Bougé, Landry Breuil, Loïc Cudennec, Sébastien Monnet.

The RESPIRE Project of the ANR MD program aims at providing a peer-to-peer (P2P) environment for advanced data management applications. It started in January 2006 and gathers research teams from the “databases” area and from the “distributed systems” area, in order to take advantage from their respective background, to have a more global view of the problem and to raise synergism. The RESPIRE Project is based on the JXTA infrastructure which provides a complete abstraction from the underlying P2P network organization (DHT, flooding, super-peer). RESPIRE services are divided into basic services (peer management, communication management, group subscribing, notification, data storage and key-based retrieval) and advanced service which rely upon basic services for data access (querying), logical clustering, collaborative work and distributed query evaluation. Part of the basic services will be provided by the JXTA infrastructure. The main actions that will be developed in the project are resource access and sharing, managing logical cluster, handling replication and automated deployment of the environment. The project started in January 2006 for 3 years. Gabriel Antoniu is the local correspondent of RESPIRE for the PARIS Project-Team. Project site: http://respire.lip6.fr/.

8.2.3. ANR SI: ANR Program on Security and Informatics

8.2.3.1. ANR SI SafeScale

Participants: Jean-Louis Pazat, Françoise André, Jérémy Buisson, Nagib Abi Fadel, Xuanhua Shi.

The *SafeScale* Project is concerned with security and safety in global ambient computing systems, e.g., computational Grid.

Partners of this project are LIPN (Coordinator), ID-IMAG, ENSTB and LMC-IMAG.

We will use our adaptive techniques (e.g., DYNACO and AFPAC) to implement application reactions to attacks. This year, we first define use cases, and study how to apply our techniques. We will also compare our approach to the work-stealing technique.

8.2.4. Inria ARC: Cooperative Research Actions

8.2.4.1. ARC COA

Participants: Jean-Louis Pazat, Françoise André, Jérémy Buisson, Christian Pérez.

The PARIS Project-Team is the leader of this project. The other partners are the *Jacquard* and the *ScalApplix* Project-Teams of INRIA Futurs.

The objective of COA (*COntrol of distributed Applications*) is to design an experimental platform for dynamic adaptation and computational steering of distributed applications. The design is studied using aspect programming and the *Fractal* Component Model.

The design of DYNACO was made using Fractal with the help of the knowledge of the *Jacquard* Project-Team. We have experimented the DYNACO framework for computational steering with the *ScalApplix* Project-Team.

8.3. European grants

8.3.1. CoreGRID

Participants: Françoise André, Gabriel Antoniu, Hinde Lilia Bouziane, Jérémy Buisson, Päivi Palosaari, Christian Pérez, Thierry Priol.

Thierry Priol is the Scientific Coordinator of a *Network of Excellence* proposal, called COREGRID, in the area of Grid and Peer-to-Peer (P2P). The COREGRID network started on September 1, 2004. As many as 42 partners, mostly from 17 European countries are involved. The COREGRID Network of Excellence aims at building a European-wide research laboratory that will achieve scientific and technological excellence in the domain of large-scale distributed, Grid, and Peer-to-Peer computing. The primary objective of the COREGRID Network of Excellence is to build solid foundations for Grid and Peer-to-Peer computing both on a methodological basis and a technological basis. This will be achieved by structuring research in the area, leading to integrated research among experts from the relevant fields, more specifically distributed systems and middleware, programming models, knowledge discovery, intelligent tools, and environments.

The research programme is structured around six complementary research areas, i.e., work packages that have been selected on the basis of their strategic importance, their research challenges, and the European expertise in these areas to develop next generation Grids: *knowledge and data management*, *programming models*, *system architecture*, *Grid information and monitoring services*, *resource management and scheduling*, *problem solving environments, tools and Grid systems*.

INRIA is managing the network in collaboration with the ERCIM office which is in charge for administrative and financial management, while Thierry Priol as a *Scientific Coordinator* (SCO) is leading the Network with regard to the scientific aspects and the overall running of the project. At the beginning of the project, he established an SCO office for him and his assistant Päivi Palosaari, who began her work on December 1, 2004. The main tasks of the SCO office during the first year were coordination and monitoring of the activities related to the scientific and technical workpackages, coordinating the COREGRID *Scientific Advisory Board*, performing the first ranking of partners activity, coordinating the preparation of the second *Joint Programme of Activities* and providing the first internal assessment of the network. In addition, the SCO office participated in dissemination tasks by giving presentations, contributing to the COREGRID Newsletters etc.

Christian Pérez is responsible for the COREGRID contract within INRIA. He is responsible for managing the four INRIA project-teams (PARIS, *Grand-Large*, *OASIS* and *SARDES*) with regard to periodic reporting, etc. His main tasks were to represent INRIA in the COREGRID Members General Assembly meetings and votes.

8.3.2. GridCoord

Participants: Luc Bougé, Thierry Priol.

The *Specific Support Action* (SSA) *ERA pilot on a co-ordinated Europe-wide initiative in Grid Research* addresses the Strategic Objective 2.3.2.8 *Grid-based Systems for solving complex problems* and the Strategic Objective 2.3.6 *General Accompanying actions* as described in the IST Work Programme 2003-04. It has been launched in July 2004 for 18 months, then extended to 24 months. It ended on June 30, 2006.

Several Grid Research initiatives have been on-going or planned at national and European Community level in the last years. These initiatives proposed the development a rich set of advanced technologies, methodologies and applications, however enhanced co-ordination among the funding bodies is required to achieve critical mass, avoid duplication and reduce fragmentation in order to solve the challenges ahead. However, if Europe wishes to compete with leading global players, it would be sensible to attempt to better coordinate its various, fragmented efforts toward achieving a critical mass and the potential for a more visible impact at an international level.

The goal of the GRIDCOORD SSA proposal was namely to enlight the tracks toward such a coordinated approach. It includes: (1) Co-ordination among the funding authorities; (2) Collaboration among the individual researchers; (3) A visionary research agenda. This project was thus tightly connected to the COREGRID Network of Excellence proposal above, led by Thierry Priol at the European level.

The GRIDCOORD SSA proposal has been led by Marco Vanneschi, University of Pisa, Italy, then Geleyn Meijer, Logica and University of Amsterdam (UvA). It included 14 institutional partners from 10 European countries. The French partners were INRIA and University of Nice Sophia-Antipolis. The Final review was held on June 20, 2006 in Brussels, and the project was successfully closed on October 18, 2006.

8.3.3. XtreemOS

Participants: Yvon Jégou, Adrien Lèbre, Sandrine L'Hermitte, David Margery, Christine Morin, Thierry Priol, Thomas Ropars, Oscar Sanchez.

Christine Morin is the *Scientific Coordinator* (SCO) of the XTREEMOS Integrated Project (IP) that addresses Strategic Objective 2.5.4 *Advanced Grid Technologies, Systems and Services*, Focus 3 on *Network-centric Grid Operating Systems* as described in the IST 2006 Work Programme.

The XTREEMOS projects aims at the design, implementation, evaluation and distribution of an open source Grid operating system with native support for virtual organizations and capable of running on a wide range of underlying platforms, from clusters to mobiles. The approach we propose in this project is to investigate the construction of a new Grid OS, XTREEMOS, based on the existing general-purpose OS Linux [65].

It is a 4-year project that started in June 2006. It involves 19 partners from 7 European countries plus China. The XTREEMOS consortium composition is a balance between academic and industrial partners interested in designing and implementing the XTREEMOS components (Linux extensions to support VOs and Grid OS services), packaging and distributing the XTREEMOS system on different hardware platforms, promoting and providing user support for the XTREEMOS system, and experimenting with Grid applications using the XTREEMOS system. Various end-users are involved in XTREEMOS project, providing a large variety of test cases in scientific and business computing domains.

INRIA is managing the project in collaboration with the *Caisse des Dépôts et Consignations* (CDC). CDC is in charge for administrative and financial management, while Christine Morin as a scientific coordinator is leading the project with regard to the scientific and technical aspects. The XTREEMOS Project Office was established at the beginning of the project for her, her assistant Sandrine L'Hermitte and the technical manager, Oscar Sanchez, who started working in October 2006. The main tasks of the Project Office during 2006 were coordination and monitoring of the project activities, ensuring the clerical support of XTREEMOS management bodies: Governing Board, Executive Committee, Scientific Advisory Committee, IPUDC, coordinating the work on the consortium agreement edition, organizing the kick-off meeting, meetings of the management bodies, general technical meetings and the first informal review. In addition, the Project Office participated in dissemination and communication tasks by giving presentations, creating the XTREEMOS internal and external web-sites (http://www.xtreemos.eu/), preparing a poster, a flyer and XTREEMOS electronic newsletter.

Jean-Pierre Banâtre is the INRIA representative at the *Governing Board*. Thierry Priol is a member of the *Scientific Advisory Committee*.

Yvon Jégou leads the WP4.3 Work-Package aiming at setting up XTREEMOS testbeds. The GRID 5000 experimental grid platform will be used as a testbed by XTREEMOS partners. Christine Morin leads WP1.1, Project management, WP2.1, Virtual Organization support in Linux, WP2.2 Federation management and WP5.3, Collaboration with other IST Grid-related projects.

8.4. International bilateral grants

8.4.1. North-America

UIUC-INRIA-CNRS In 2005 we started a 2-year collaboration with Indranil Gupta's team from the University of Illinois at Urbana Champaign (UIUC). Indranil Gupta visited the PARIS Research Group for one week in June 2006. We worked on alternative techniques for P2P management of large-scale groups and we defined an application-driven peer-to-peer overlay. This work has been published at SRDS [40].

Rutgers University, USA. We have collaborated with the *Discolab* Research Team leaded by Liviu Iftode at Rutgers University in the framework of the PHENIX Associated Team funded by INRIA since January 2005. We investigate the design of a novel, highly-available cluster architecture based on the concept of *remote healing*. Benoît Boissinot, a Master student from ENS Lyon, performed his Master internship [56] in the PARIS Project-Team on distributed system monitoring and failure diagnosis using co-operative virtual backdoors in the framework of the PHENIX Associated Team. He spent 10 weeks at Rutgers University in the period February–April 2006. Christine Morin and Pascal Gallard visited Rutgers University in April 2006 (two days). Liviu Iftode visited IRISA for two days in June 2006 and attended Benoît Boissinot's internship defense at ENS Lyon. The second PHENIX Workshop on Global Computing was organized at IRISA on December 7 and 8, 2006 (http://www.irisa.fr/paris/web/phenix-ws-2006.html). Liviu Iftode and two of his students participated in this workshop.

8.4.2. Middle-East, Asia, Oceania

INRIA-AIST In 2006 we started a two-year bilateral collaboration with AIST (Tsukuba, Japon). The goal is to study interactions between the JUXMEM grid data sharing service developed by the PARIS Project-Team and the *GFarm global file system* developed at AIST. Then, one could build a common data sharing infrastructure providing the ability to share *both* RAM memory and secondary storage. Gabriel Antoniu and Loïc Cudennec visited the AIST team for one week in October 2006 and Osamu Tatebe (University of Tsukuba) visited the PARIS Project-Team for one week in December 2006.

9. Dissemination

9.1. Community animation

9.1.1. Leaderships, Steering Committees and community service

European COREGRID IST-FP6 Network of Excellence. Th. Priol is the *Scientific Coordinator* of the COREGRID Network of Excellence (http://www.coregrid.net/). This network started on September 2004, for a duration of four years. Ch. Pérez is the INRIA Scientific Correspondent of COREGRID NoE.

ACI GRID, Ministry of Research. Th. Priol is the Director of the ACI GRID Program, funded by the French National Ministry of Research. The ACI GRID is the national French initiative in the area of Grid computing.

CNRS, GDR ASR. J.-L. Pazat is co-director of the GSP working group on Grids, Systems and Parallelism of the CNRS Research Co-operative Federation (*Groupement de recherche*, GDR) ASR (*Architectures, Systems and Networks*). F. André serves as the coordinator of the Action ADAPT (*Dynamic Adaptation*) of the GSP working group.

RenPar Conference Series J.-L. Pazat serves as the Chair of the Steering Committee of the RenPar annual conference series (*Rencontres francophone du parallélisme*, http://www.renpar.org/). The last edition of RenPar was held in Perpignan in 2006.

Euro-Par Conference Series. L. Bougé serves as the Vice-Chair of the *Steering Committee* of the *Euro-Par* annual conference series on parallel computing (around 250 attendees, http://www.europar.org/).

IEEE IPDPS Conference Series. L. Bougé is a member of the *Steering Committee* of the IPDPS annual conference series (*International Parallel and Distributed Processing Symposium*, http://www.ipdps.org/) .

European GRIDCOORD IST-FP6 SSA. L. Bougé and Th. Priol participated to the GRIDCOORD *Specific Support Action* (SSA) through the INRIA institutional member. After the tragic passing away of Isabelle Attali, INRIA Sophia-Antipolis, L. Bougé was in charge of leading the contribution of INRIA members to this SSA, in close co-ordination with the COREGRID NoE. The 2-year GRIDCOORD SSA successfully ended in June 2006 (http://www.gridcoord.org/).

ANR MD GDS. G. Antoniu headed the GDS (*Grid Data Service*) Project supported by the ANR MD Program on Data Masses. This 3-year project ended in September 2006 (http://www.irisa.fr/GDS/).

ANR MD GdX. G. Antoniu was the local correspondent of the GdX (*Data Grid Explorer*) Project supported by ANR MD Program on Data Masses. This 3-year project ended in September 2006 (http://www.lri.fr/~fci/GdX/).

ANR MD RESPIRE. G. Antoniu is the local correspondent of the RESPIRE Project (*Peer-to-peer resources and services, querying and replication*). This 3-year project started in January 2006 (http://respire.lip6.fr/).

Agrégation of Mathematics. L. Bougé serves as one of the Vice-Chairs of the national selection competition for high-school mathematics teachers (*Agrégation de mathématiques*). He is in charge of the newly-founded *Fundamental Computer Science* track of the selection competition.

ANR CI NUMASIS. Ch. Pérez is the local correspondent of the NUMASIS Project (*Adaptation et optimisation des performances applicatives sur architectures NUMA. Étude et mise en oeuvre sur des applications en SISmologie*). This 3-year project started in January 2006 (http://numasis.gforge.inria.fr/).

ANR CI DISC. Ch. Pérez is the local correspondent of the DISC Project (*Distributed objects and components for high performance scientific computing on the* GRID 5000 *test-bed*). This 3-year project started in January 2006 (http://www-sop.inria.fr/caiman/personnel/Stephane.Lanteri/discogrid/).

ANR CI LEGO. G. Antoniu is the local correspondent of the LEGO Project (*League for Efficient Grid Operation*). This 3-year project started in January 2006 (http://graal.ens-lyon.fr/LEGO/).

European XTREEMOS IST-FP6 Integrated Project. Christine Morin is the *Scientific Coordinator* of the XTREEMOS Integrated Project (http://www.xtreemos.eu/). This integrated project started on June 2006, for a duration of four years. Yvon Jégou is a member of XTREEMOS Executive Committee. Thierry Priol is a member of XTREEMOS Scientific Advisory Committee. Jean-Pierre Banâtre is the INRIA representative in the XTREEMOS Governing Board.

9.1.2. Editorial boards, direction of program committees

L. Bougé is a member of the *Editorial Advisory Board* of the *Scientific Programming* Journal, IOS Press.

Th. Priol is a member of the Editorial Board of the *Parallel Computing* Journal.

He is a member of the Editorial Board of the *International Journal of Web Services Research*.

He was Vice-Chair of the Program Committee of the *2nd IEEE International Conference on e-Science and Grid Computing*, Amsterdam, The Netherlands, December 2006.

He was Vice-Chair (Track *Information System and Grid Technologies*) of the Program Committee of the *IEEE 2006 John Vincent Atanasoff International Symposium on Modern Computing*, Sophia, Bulgaria, October 2006.

Ch. Morin served as the Vice-Chair of Topic 8 on Distributed Systems and Algorithms at the *Euro-Par 2006* Conference, Dresden, Germany, August 2006.

9.1.3. Program Committees

Th. Priol served in the Program Committees of the following conferences:

- CCGRID 2006: *IEEE International Symposium on Cluster Computing and the Grid*, Singapore, May 2006.
- DAPSYS 2006: *6th Austrian-Hungarian Workshop on Distributed and Parallel Systems*, Innsbruck, Austria, September 2006.
- EXPGRID 2006: *Workshop on Experimental Grid Testbed for the Assessment of Large-Scale Distributed Applications and Tools*. In conjunction with HPDC 2006. Paris, France, June 2006.
- GADA 2006: *International Conference on Grid computing, high-PerformAnce and Distributed Applications*. Montpellier, France, November 2006.
- GRID 2006: *7th IEEE/ACM Intl. Conference on Grid Computing*, Barcelona, Spain, September 2006.
- HPC-GECO/CompFrame 2006: *Workshop on Programming Environments and COmponents*. In conjunction with HPDC 2006. Paris, France June 2006.
- HPDC 2006: *15th IEEE International Symposium on High-Performance Distributed Computing*, Paris, France, June 2006.
- HPGC 2006: *Workshop on High Performance Grid Computing*. In conjunction IPDPS 2006. Rhodes Island, Greece, April 2006.
- ICWS 2006: *IEEE International Conference on Web Services*, Chicago, Illinois, September 2006.
- WI 2006: *The IEEE/WIC/ACM International Conference on Web Intelligence, and the International Conference on Intelligent Agent Technology*, Hong-Kong, China, December 2006.
- WDMG 2006: *VLDB Workshop on Data Management in Grids*, Seoul, Korea, September 2006.
- VECPAR 2006: *The 7th International Meeting on high performance computing for computational science*, Rio de Janeiro, Brasil, July 2006.

C. Morin served in the Program Committees of the following conferences:

- Cluster 2006: *IEEE Cluster 2006*, Barcelona, Spain, September 2006.
- DSM 2006: *Sixth International Workshop on Distributed Shared Memory*, held in conjunction with CCGRID 2006, Singapore, May 2006.
- ICPP 2006: *35th International Conference on Parallel Processing*, Colombus, Ohio, August 2006.
- ISPA 2006: *Fourth International Symposium on Parallel and Processing and Applications (ISPA'06)*, Sorrento, Italy, December 2006.
- RenPar 2006: *Rencontres francophones du Parallélisme*, Perpignan, France, October 2006.

Ch. Pérez served in the Program Committees of the following conferences:

- HPC-GECO/CompFrame 2006: *Joint Workshop on HPC Grid programming Environments and COmponents and Component and Framework Technology in High-Performance and Scientific Computing*, Paris, June 2006.
- GELA 2006: *Workshop on Grid-Enabling Legacy Applications and Supporting End Users Workshop*, Paris, June 2006.
- RenPar 17: 17e Rencontres francophones du parallélisme, Perpignan, France, October 2006.
- LESGAD 2006: *Workshop on Languages and Execution Support for Grid Application Development*, Workshop 15 of the 20th edition of the European Conference on Object-Oriented Programming (ECOOP 2006), Nantes, France, July 2006.

G. Antoniu served in the Program Committees for the following conferences:

ICPP 2006: *International Conference on Parallel Processing (ICPP 2006)*, Colombus, USA, August 2006.

CCGrid 2006: *IEEE/ACM International Symposium on Cluster Computing and the Grid*, Singapore, May 2006.

PDMST 2006: *International Workshop on P2P Data Management, Security and Trust*, Krakow, Poland, September 2006, in conjunction with DEXA 2006.

HPDGrid 2006: *International Workshop on High-Performance Data Management*, Rio de Janeiro, Brazil, July 2006, in conjunction with VecPar 2006.

9.1.4. Evaluation committees, consulting

L. Bougé served in the 2006 International Assessment Panel of the FOCUS/BRICKS program of the Dutch National Science Foundation (NWO).

He was solicited as an expert by the French Ministry of Higher Education (MSTP STIC) for the evaluation of a doctoral school.

He was also solicited as an expert by a French Region to evaluate applications for PhD grants.

C. Morin acted as a referee for the Foreign PhD Committee of Martin Kacer from CTU, Prague, Czech Republic.

Th. Priol is a member of the Scientific Committee of the ANR CI Program on *High Performance Computing and Simulation* of the French National Research Agency.

He is a member of the Scientific Committee of the PRST Intelligence Logicielle (*Contrat de Plan Etat-Région Lorraine 2003-2006*).

He was solicited as an expert for the CEA/DAM Scientific Committee to review research activities related to high-performance software environment.

He was an evaluator for The Canada Foundation for Innovation.

Ch. Pérez was solicited as an expert for the ANR CI Program on *High Performance Computing and Simulation* and the ANR MD Program on *Data Masses* of the French National Research Agency.

J.-L. Pazat was solicited as an expert for theANR CI Program on *High Performance Computing and Simulation* of the French National Research Agency.

9.2. Academic teaching

G. Antoniu is teaching part of the *Operating Systems* Module at *IUP 2 MIAGE*, IFSIC. He has given lectures on peer-to-peer systems within the *High Performance Computing on Clusters and Grids* Module and within the *Peer-to-Peer Systems* Module of the Master Program, UNIVERSITY RENNES 1, and within the *Distributed Systems* Module taught for the final year engineering students of INSA Rennes.

A. Lèbre gave lectures on high performance I/O in clusters within the *High Performance Computing on Clusters and Grids* Module of the Master Program, UNIVERSITY RENNES 1.

Ch. Morin is responsible for a graduate teaching Module *High Performance Computing on Clusters and Grids* of the Master Program in Computer Science, UNIVERSITY RENNES 1. Within this module, she gave lectures on distributed operating systems for clusters.

Th. Priol gave lectures on Distributed Shared Memory within the *High Performance Computing on Clusters and Grids* Module of the Master Program, UNIVERSITY RENNES 1.

Ch. Pérez gave lectures to 5th-year students of INSA of Rennes on CORBA and CCM within the course *Objects and components for distributed programming*.

He also gave lectures to 5th-year students of Polytech Nantes on CORBA and CCM within the course *Objects and components for distributed programming*.

J.-L. Pazat leads the Master Program of the 5th year of Computer Science at INSA of Rennes. He is responsible for a teaching module on Parallel Processing for engineers at INSA of Rennes. Within this module, he gave lectures on parallel and distributed programming. He is responsible for a graduate teaching module Objects and components for distributed programming for 5th-year students of INSA of Rennes.

9.3. Conferences, seminars, and invitations

Only the events not listed elsewhere are listed below.

L. Bougé gave a lecture on Grid Computing at the *30th Anniversary Open Days* of IRISA. He also gave this lecture for a high-school class at the *Rennes Month of Science*.

B. Boissinot was invited to present a talk entitled *Distributed System Monitoring and Failure Diagnosis using Cooperative Virtual Backdoors* at the first Rutgers/Pierre et Marie Curie Joint Workshop, Paris, June 2006.

E. Jeanvoine presented a talk entitled *Resource Allocation in Large Scale Grids* at the GRID 5000 Winter school, Aussois, March 2006.

E. Jeanvoine presented a talk entitled *Resource Management in Large Scale Grids* at the meeting of the SINETICS group, EDF R&D, Clamart, March 2006.

E. Jeanvoine was invited to present a talk entitled *Allocation de Ressources dans les Grilles*, at the GdX days, Orsay, December 2006.

R. Lottiaux was invited to present a talk on KERRIGHED at Linux Solutions 2006, Paris, February 2006.

R. Lottiaux and P. Gallard participated to *Tremplin de la recherche*, an event organized by the French *Sénat*, to present the KERLABS start-up created to valorize KERRIGHED cluster operating system, Paris, February 2006.

R. Lottiaux was invited to present a talk entitled *The* KERRIGHED *operating system: an overview* at the *Observatoire astronomique de Strasbourg*, March 2006.

C. Morin was invited to participate as a panelist on *Non-US Issues in Supercomputing* to the SOS 10 Workshop on Distributed Supercomputing held in Maui, Hawaii, USA, March 2006.

C. Morin was invited to give a talk entitled *Toward Grid-aware Operating Systems* at ORNL, Oak Ridge, USA, April 2006.

C. Morin was invited to present the XTREEMOS Integrated Project in the *Powering the Grid* Session at the public launch of the new IST Grid projects event during the *European Grid Technology Days*, September, 2006.

C. Morin was invited to present her experience with the XTREEMOS Integrated Project at the INRIA seminar for new-comers, Roissy, November 2006.

C. Morin was invited to present her experience in the preparation of XTREEMOS Integrated Project proposal at the European Framework Programme 7 in Brittany Days organized by the *Noé Regional Network* and the Regional Council of Brittany, Rennes, November 2006.

J. Parpaillon has presented a talk entitled *SSI Deployment Issues* at ORNL, Oak Ridge Tennessee, in April 2006.

J. Parpaillon has presented a tutorial on KERRIGHED deployment in the framework of a training session organized for XTREEMOS participants, IRISA, Rennes, October 2006.

J. Parpaillon presented a talk on SSI-OSCAR at the OSCAR BOF during SC '06, Tampa, USA, November 2006.

CGW 2006. Th. Priol gave a keynote presentation on the ACI GRID Program and the GRID 5000 Project in Cracow, Poland, October 2006.

IST 2006. Th. Priol gave an invited presentation on *From Grids to service-oriented knowledge utilities - research challenges* at the IST 2006 Session on *service and software architectures, infrastructures and engineering*.

L. Rilling has presented a talk entitled *Reliable Execution of Distributed Applications Sharing Data in a Single Image Grid Operating System*, at the Parallélisme/Grand-Large seminar, Orsay, January 2006

9.4. Administrative responsibilities

F. André is the vice-chair of the Administrative Committee of IFSIC, the Computer Science department of UNIVERSITY RENNES 1.

L. Bougé chairs the Computer Science and Telecommunication Department (*Département Informatique et Télécommunications, DIT*) of the Brittany Extension of ENS CACHAN on the Ker Lann Campus in Bruz, in the close suburb of Rennes.

He leads the Master Program in Computer Science at the Brittany Extension of ENS CACHAN (*Magistère Informatique et Télécommunications*, for short, the famous MIT Rennes :-)). This program is co-supported with UNIVERSITY RENNES 1. It was launched in September 2002. Olivier Ridoux, LANDE Project-Team, IRISA, co-supervises the program for UNIVERSITY RENNES 1.

He serves as the Vice-Chairman of the Selection Committee (*Commission de spécialistes d'Établissement*, CSE) for Computer Science at ENS CACHAN, and as an external deputy-member of the Computer Science CSE at UNIVERSITY RENNES 1.

Ch. Morin chaired the local IRISA Computing Infrastructure User Committee (*Commission des utilisateurs des moyens informatiques*, CUMI) till March 2006.

She is an external member of the *Course Advisory Board* of the Information Technology School of Deakin University (Australia).

J.-L. Pazat is a member of the Computer Science Department committee.

9.5. Miscellaneous

F. André is a member of the Selection Committee (*Commission de spécialistes*, CSE) of IFSIC (Computer Science department of UNIVERSITY RENNES 1), of the Computer Science department of INSA of Rennes and of the Computer Science group of University of Rennes 2.

L. Bougé is a member of the Project-Team Committee of IRISA, standing for the ENS CACHAN partner.

Ch. Morin was a member of the Editorial Board of *Inedit*, the INRIA Newsletter till September 2006.

Ch. Pérez is member of the Selection Committee (*Commission de spécialistes*, CSE) for Computer Science at ENS CACHAN. He is a member of the IRISA Committee (*Conseil de laboratoire*).

T. Priol is a member of the Project-Team Committee of IRISA.

J.-L. Pazat is a member of the Selection Committee (*Commission de spécialistes*, CSE) for Computer Science at INSA Rennes.

10. Bibliography

Major publications by the team in recent years

[1] M. ALDINUCCI, F. ANDRÉ, J. BUISSON, S. CAMPA, M. COPPOLA, M. DANELUTTO, C. ZOCCOLO. *Parallel program/component adaptivity management*, in "ParCo 2005, Málaga, Spain", 13-16 September 2005, http://www.irisa.fr/paris/Biblio/Papers/Buisson/AldAndBuiCamCopDanZoc05PARCO.pdf.

[2] F. André, M. Le Fur, Y. Mahéo, J.-L. Pazat. *The Pandore Data Parallel Compiler and its Portable Runtime*, in "High-Performance Computing and Networking (HPCN Europe 1995), Milan, Italy", Lecture Notes in Computer Science, vol. 919, Springer Verlag, May 1995, p. 176–183.

[3] G. Antoniu, L. Bougé. *DSM-PM2: A portable implementation platform for multithreaded DSM consistency protocols*, in "Proc. 6th International Workshop on High-Level Parallel Programming Models and Supportive Environments (HIPS '01), San Francisco", Lect. Notes in Comp. Science, Available as INRIA Research Report RR-4108, vol. 2026, Springer-Verlag, Held in conjunction with IPDPS 2001. IEEE TCPP, April 2001, p. 55–70, http://hal.inria.fr/inria-00072523.

[4] J.-P. Banâtre, D. Le Métayer. *Programming by Multiset Transformation*, in "Communications of the ACM", vol. 36, n$^{\text{O}}$ 1, January 1993, p. 98–111.

[5] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, A. Singh. *SplitStream: High-Bandwidth Multicast in Cooperative Environments*, in "Symposium on Operating System principles (SOSP 2003), Bolton Landing, NY", October 2003, http://www.irisa.fr/paris/Biblio/Papers/Kermarrec/CasDruKerNanRowSin03SOSP.pdf

[6] A. Denis, C. Pérez, T. Priol. *PadicoTM: An Open Integration Framework for Communication Middleware and Runtimes*, in "IEEE Intl. Symposium on Cluster Computing and the Grid (CCGrid2002), Berlin, Germany", Available as INRIA Reserach Report RR-4554, IEEE Computer Society, May 2002, p. 144–151, http://hal.inria.fr/inria-00072034.

[7] P. Eugster, P. Felber, R. Guerraoui, A.-M. Kermarrec. *The Many Faces of Publish/Subscribe*, in "ACM computing Surveys", vol. 35, n$^{\text{O}}$ 2, June 2003, http://www.irisa.fr/paris/Biblio/Papers/Kermarrec/EugFelGueKer03ACMSur

[8] A.-M. Kermarrec, C. Morin, M. Banâtre. *Design, Implementation and Evaluation of ICARE*, in "Software Practice and Experience", n$^{\text{O}}$ 9, 1998, p. 981–1010.

[9] T. Kielmann, P. Hatcher, L. Bougé, H. Bal. *Enabling Java for High-Performance Computing: Exploiting Distributed Shared Memory and Remote Method Invocation*, in "Communications of the ACM", Special issue on Java for High Performance Computing, vol. 44, n$^{\text{O}}$ 10, October 2001, p. 110–117.

[10] Z. Lahjomri, T. Priol. *KOAN: A Shared Virtual Memory for iPSC/2 Hypercube*, in "Proc. of the 2nd Joint Int'l Conf. on Vector and Parallel Processing (CONPAR'92)", Lecture Notes in Computer Science, vol. 634, Springer Verlag, September 1992, p. 441–452, http://hal.inria.fr/inria-00074927.

[11] T. Priol. *Efficient support of MPI-based parallel codes within a CORBA-based software infrastructureResponse to the Aggregated Computing RFI from the OMG, Document orbos/99-07-10*, July 1999.

Year Publications

Books and Monographs

[12] S. Gorlatch, M. Bubak, T. Priol (editors). *Integrated Research in Grid Computing*, n$^{\text{O}}$ ISBN 83-915141-6-1, Academic Computer Centre CYFRONET AGH, 2006.

Doctoral dissertations and Habilitation theses

[13] J. Buisson. *Adaptation dynamique de programmes et composants parallèles*, Thèse de doctorat, INSA de Rennes, IRISA, Rennes, France, September 2006.

[14] M. JAN. *JuxMem : un service de partage transparent de données pour grilles de calculs fondé sur une approche pair-à-pair*, Thèse de doctorat, Université de Rennes 1, IRISA, Rennes, France, November 2006.

[15] S. MONNET. *Gestion des données dans les grilles de calcul : support pour la tolérance aux fautes et la cohérence des données*, Thèse de doctorat, Université de Rennes 1, IRISA, Rennes, France, November 2006.

Articles in refereed journals and book chapters

[16] G. ANTONIU, M. BERTIER, E. CARON, F. DESPREZ, L. BOUGÉ, M. JAN, S. MONNET, P. SENS. *Future Generation Grids*, CoreGRID series, chap. GDS: An Architecture Proposal for a grid Data-Sharing Service, Springer Verlag, 2006, p. 133-152.

[17] G. ANTONIU, J.-F. DEVERGE, S. MONNET. *How to bring together fault tolerance and data consistency to enable grid data sharing*, in "Concurrency and Computation: Practice and Experience", vol. 18, n^{o} 13, November 2006, p. 1705–1723, http://hal.inria.fr/inria-00000987.

[18] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *Generalised multisets for chemical programming*, in "Mathematical Structures in Computer Science", vol. 16, n^{o} 4, August 2006, p. 557–580.

[19] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *Programming Self-Organizing Systems with the Higher-Order Chemical Language*, in "International Journal of Unconventional Computing (IJUC)", to appear, 2006.

[20] A. BOUTEILLER, H. L. BOUZIANE, T. HÉRAULT, P. LEMARINIER, F. CAPPELLO. *Hybrid Preemptive Scheduling of MPI Applications on the Grids*, in "In Int. Journal of High Performance Computing Special issue", (IJHPCA), vol. 20, n^{o} 1, 2006, p. 77-90.

[21] J. BUISSON, F. ANDRÉ, J.-L. PAZAT. *Afpac: Enforcing consistency during the adaptation of a parallel component*, in "Scalable Computing: Practice and Experience", electronic journal (http://www.scpe.org/). extended version of citeBuiAndPaz05ISPDC., vol. 7, n^{o} 3, September 2006, p. 83–95, http://www.scpe.org/vols/vol07/no3/SCPE_7_3_06.pdf.

[22] O. MICHEL, J.-P. BANÂTRE, J.-L. GIAVITTO, P. FRADET. *Challenging Questions for the Rationals of Non-Classical Programming Languages*, in "International Journal of Unconventional Computing (IJUC)", vol. 2, n^{o} 4, 2006.

Publications in Conferences and Workshops

[23] N. ABI FADEL. *Adaptation Dynamique d'applications parallèles*, in "Actes des Rencontres francophones du parallélisme (RenPar 17), Canet-en-Roussillon", October 2006, p. 76–83.

[24] M. ALDINUCCI, F. ANDRÉ, J. BUISSON, S. CAMPA, M. COPPOLA, M. DANELUTTO, C. ZOCCOLO. *An Abstract Schema Modelling Adaptivity Management*, in "CoreGrid Integration Workshop", CoreGRID proceedings, vol. 4, Springer, 2006, http://www.irisa.fr/paris/Biblio/Papers/Buisson/AldAndBuiCamCopDanZoc06CGIW.pdf.

[25] M. ALDINUCCI, G. ANTONIU, M. DANELLUTO, M. JAN. *Fault-tolerant data sharing for high-level grid programming: a hierarchical storage achitecture*, in "Proc. CoreGrid Integration Workshop, Krakow, Poland", October 2006, p. 177-188.

[26] G. ANTONIU, H. L. BOUZIANE, L. BREUIL, M. JAN, C. PÉREZ. *Enabling Transparent Data Sharing in Component Models*, in "6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID), Singapore", May 2006, p. 430-433, https://hal.inria.fr/inria-00101363/en/.

[27] G. ANTONIU, H. L. BOUZIANE, M. JAN, C. PÉREZ, T. PRIOL. *Combining Data Sharing with the Master-Worker Paradigm in the Common Component Architecture*, in "Proc. Joint Workshop on HPC Grid programming Environments and COmponents and Component and Framework Technology in High-Performance and Scientific Computing (HPC-GECO/CompFrame 2006), Paris, France", Held in conjunction with HDPC-15, June 2006, p. 10-18, https://hal.inria.fr/inria-00101365/en/.

[28] G. ANTONIU, L. CUDENNEC, S. MONNET. *A practical evaluation of a data consistency protocol for efficient visualization in grid applications*, in "International Workshop on High-Performance Data Management in Grid Environment (HPDGrid 2006), Rio de Janeiro, Brazil", Selected for publication in the post-conference book, Held in conjunction with VECPAR'06, July 2006.

[29] G. ANTONIU, L. CUDENNEC, S. MONNET. *Extending the entry consistency model to enable efficient visualization for code-coupling grid applications*, in "6th IEEE/ACM International Symposium on Cluster Computing and the Grid, Singapore", CCGrid 2006, May 2006, p. 552-555.

[30] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *A Generalized Higher-Order Chemical Computation Model*, in "International Workshop on Developments in Computational Models", ENTCS, vol. 135, n^{o} 3, Elsevier, March 2006, p. 3–13, http://www.cs.york.ac.uk/nature/workshop/papers/BanatreFradetRadenac.pdf.

[31] H. L. BOUZIANE, C. PÉREZ. *Du support du paradigme maître-travailleur dans les modèles de composants logiciels*, in "Journées Composants (JC 2006), Perpignan, France", October 2006.

[32] H. L. BOUZIANE, C. PÉREZ, N. CURRLE-LINDE, M. RESCH. *A Software Component-based Description of the SEGL Runtime Architecture*, in "CoreGrid integration workshop 2006, Krakow, Poland", October 2006, p. 69-80.

[33] H. L. BOUZIANE, C. PÉREZ, T. PRIOL. *Modeling and executing Master-Worker applications in component models*, in "11th International Workshop on High-Level Parallel Programming Models and Supportive Environments (HIPS), Rhodes Island, Greece", April 2006, http://www.irisa.fr/paris/pages-perso/Hinde-Lilia-Bouziane/bibliography/master-workerInCompMod06.ps.

[34] J. BUISSON, F. ANDRÉ, J.-L. PAZAT. *Performance and Practicability of Dynamic Adaptation for Parallel Computing*, in "The 15th IEEE International Symposium on High Performance Distributed Computing, Paris, France", poster session., June 2006, p. 331-332, http://www.irisa.fr/paris/Biblio/Papers/Buisson/BuiAndPaz06HPDC.pdf.

[35] E. JEANVOINE, C. MORIN, D. LEPRINCE. *Un protocole de découverte de ressources optimisé pour l'allocation de ressources dans les grilles*, in "Actes de CFSE-5 2006, Perpignan, France", In french, October 2006.

[36] E. JEANVOINE, L. RILLING, C. MORIN, D. LEPRINCE. *Architecture distribuée pour la gestion des ressources dans des grilles à grande échelle*, in "Actes de NOTERE 2006, Toulouse, France", In french, June 2006, http://hal.inria.fr/inria-00070210.

[37] E. JEANVOINE, L. RILLING, C. MORIN, D. LEPRINCE. *Using Overlay Networks to Build Operating System Services for Large Scale Grids*, in "Proceedings of the 5th International Symposium on Parallel and Distributed Computing (ISPDC 2006), Timisoara, Romania", July 2006, http://hal.inria.fr/inria-00070244.

[38] Y. JÉGOU. *Dynamic Memory Management on Mome DSM*, in "Proc. Intl. Workshop on Distributed Shared Memory on Clusters (DSM 2006), Singapore", Held in conjunction with CCGrid 2006, May 2006.

[39] S. MONNET, M. BERTIER. *Using failure injection mechanisms to experiment and evaluate a grid failure detector*, in "Workshop on Computational Grids and Clusters (WCGC 2006), Rio de Janeiro, Brazil", Selected for publication in the post-conference book, Held in conjunction with VECPAR'06, July 2006.

[40] S. MONNET, R. MORALES, G. ANTONIU, I. GUPTA. *MOve: Design of An Application-Malleable Overlay*, in "Symposium on Reliable Distributed Systems 2006 (SRDS 2006), Leeds, UK", IEEE Computer Society, October 2006, p. 355-364.

[41] Z. NÉMETH, C. PÉREZ, T. PRIOL. *Distributed Workflow Coordination: Molecules and Reactions*, in "The 9th International Workshop on Nature Inspired Distributed Computing (NIDISC'06)", April 2006.

[42] L. RILLING. *Vigne: Towards a Self-Healing Grid Operating System*, in "Proceedings of Euro-Par 2006, Dresden, Germany", Lecture Notes in Computer Science, vol. 4128, Springer, August 2006, p. 437-447, http://www.irisa.fr/paris/web/images/stories/lrilling/Vigne_self_healing.pdf.

[43] L. RILLING. *Vigne : vers un système d'exploitation auto-réparant pour la grille*, in "Actes de la 5ème Conférence Française sur les Systèmes d'Exploitation (CFSE 5), Perpignan, France", October 2006.

Internal Reports

[44] G. ANTONIU, H. L. BOUZIANE, L. BREUIL, M. JAN, C. PÉREZ. *Enabling Transparent Data Sharing in Component Models*, Submitted for publication, Research Report, n^o RR-5796, INRIA, IRISA, Rennes, France, November 2006, http://hal.inria.fr/inria-00070227.

[45] G. ANTONIU, E. CARON, F. DESPREZ, M. JAN. *Towards a Transparent Data Access Model for the GridRPC Paradigm*, Research Report, n^o 6009, INRIA, November 2006, https://hal.inria.fr/inria-00110967.

[46] G. ANTONIU, E. CARON, F. DESPREZ, M. JAN. *Towards a Transparent Data Access Model for the GridRPC Paradigm*, Research Report, n^o 6009, INRIA, November 2006, https://hal.inria.fr/inria-00110967.

[47] G. ANTONIU, L. CUDENNEC, S. MONNET. *Extending the entry consistency model to enable efficient visualization for code-coupling grid applications*, Research Report, n^o RR-5813, INRIA, IRISA, Rennes, France, January 2006, http://hal.inria.fr/inria-00070211.

[48] H. L. BOUZIANE, C. PÉREZ, N. CURRLE-LINDE, M. RESCH. *A Software Component-based Description of the SEGL Runtime Architecture*, Technical report, n^o 0054, CoreGrid Network of Excellence, 2006, http://www.coregrid.net/mambo/images/stories/TechnicalReports/tr-0054.pdf.

[49] L. CUDENNEC, S. MONNET. *Extension du modèle de cohérence à l'entrée pour la visualisation dans les applications de couplage de codes sur grille*, Research report, n^o RR-5812, INRIA, IRISA, Rennes, France, January 2006, http://hal.inria.fr/inria-00070212.

[50] E. JEANVOINE, L. RILLING, C. MORIN, D. LEPRINCE. *Distributed Architecture for Resource Management in Large Scale Grids*, Research Report, n^{o} RR-5814, INRIA, IRISA, Rennes, France, January 2006, http://hal.inria.fr/inria-00070210.

[51] S. MONNET, M. BERTIER. *Using failure injection mechanisms to experiment and evaluate a hierarchical failure detector*, Research report, n^{o} RR-5811, INRIA, Rennes, France, January 2006, http://hal.inria.fr/inria-00070213.

[52] S. MONNET, R. MORALES, G. ANTONIU, I. GUPTA. *MOve: Design of an Application-Malleable Overlay*, Research Report, n^{o} RR-5872, INRIA, IRISA, Rennes, France, March 2006, http://hal.inria.fr/inria-00070154.

[53] T. ROPARS, E. JEANVOINE, C. MORIN. *Providing QoS in a Grid Application Monitoring Service*, Research Report, n^{o} RR-6070, IRISA/Paris Research group, Université de Rennes 1, EDF R&D, INRIA, IRISA, Rennes, France, December 2006, http://hal.inria.fr/inria-00121059.

Miscellaneous

[54] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *Chemical Programming of Self-Organizing Systems*, January 2006, ERCIM News 64.

[55] A. BELOUED, J.-M. GILLIOT, M.-T. SEGARRA, F. ANDRÉ. *Context-aware replication and consistency*, art. no. 0602-o2003, 2006, IEEE Distributed Systems Online, Work-in-Progress series, Vol. 7(2).

[56] B. BOISSINOT. *Distributed System Monitoring and Failure Diagnosis using Cooperative Virtual Backdoors*, Technical report, ENS Lyon, June 2006, http://www.irisa.fr/paris/bibadmin/uploads/pdf/Benoit_Boissinot_3.pdf.

[57] J. GALLARD. *Écoulement de la charge sur le système à image unique Kerrighed : application au domaine de la biologie.*, Technical report, ENSSAT - Université de Rennes 1, June 2006, http://www.irisa.fr/paris/bibadmin/uploads/pdf/jgallard.pdf.

[58] P. GALLARD, R. LOTTIAUX, C. MORIN. *Kerrighed User Manual*, March 2006, Livrable PEA COCA .

[59] P. GALLARD, R. LOTTIAUX, C. MORIN. *Kerrighed V2.0 – Etude de la problématique de haute disponibilité dans le système Kerrighed*, March 2006, Livrable PEA COCA.

[60] P. GALLARD, R. LOTTIAUX, C. MORIN. *Kerrighed V2.0 – Manuel de référence*, March 2006, Livrable PEA COCA.

[61] P. GALLARD, R. LOTTIAUX, C. MORIN. *Rapport de synthèse - Conception d'un système d'exploitation pour grappe de PCs*, March 2006, Livrable PEA COCA- Marché 02.34.058.00.470.75.65.

[62] R. GUZIOLOWSKI. *Evaluation of Kerrighed cluster operating system for the execution of Internet services*, February 2006, http://www.irisa.fr/paris/bibadmin/uploads/pdf/FinalReport_stage_RobertGuziolowski.pdf, Report on the internship at IRISA.

[63] T. PRIOL. *Un superordinateur mondial à la demande*, January 2006, http://www.larecherche.fr/special/accueil/sup393.html, La recherche - Supplément Le Calcul Haute Performance.

[64] T. ROPARS. *Supervision d'applications sur grille de calcul*, Technical report, Université de Rennes 1, June 2006, http://www.irisa.fr/paris/bibadmin/uploads/pdf/rapport_Supervision_Application.pdf.

[65] XTREEMOS CONSORTIUM. *Annex 1 - Description of Work*, XtreemOS Integrated Project, IST-033576, April 2006, Contract funded by the European Commission.

[66] XTREEMOS CONSORTIUM. *Linux-XOS Specification*, November 2006, XtreemOS Integrated Project Deliverable D2.1.1.

[67] XTREEMOS CONSORTIUM. *Specification of federation resource management mechanisms*, November 2006, XtreemOS Integrated Project Deliverable D2.2.1.

References in notes

[68] I. FOSTER, C. KESSELMAN (editors). *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Publishers, 1998.

[69] *Project JXTA: Java programmers guide*, Sun Microsystems, Inc., 2001, http://www.jxta.org/white_papers.html.

[70] *OpenMP Fortran Application Program Interface*, Version 2.0, November 2000.

[71] *Wireless Application Protocol 2.0: technical white paper*, January 2002, http://www.wapforum.org/what/WAPWhite_Paper1.pdf.

[72] R. ARMSTRONG, D. GANNON, A. GEIST, K. KEAHEY, S. KOHN, L. MCINNES, S. PARKER, B. SMOLINSKI. *Toward a Common Component Architecture for High-Performance Scientific Computing*, in "Proceeding of the 8th IEEE International Symposium on High Performance Distributed Computation", August 1999.

[73] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *Principles of Chemical Programming*, in "Proceedings of the 5th International Workshop on Rule-Based Programming (RULE 2004)", S. ABDENNADHER, C. RINGEISSEN (editors). , ENTCS, vol. 124, n^o 1, Elsevier, June 2005, p. 133–147.

[74] J.-P. BANÂTRE, P. FRADET, D. LE MÉTAYER. *Gamma and the Chemical Reaction Model: Fifteen Years After*, in "Multiset Processing", LNCS, vol. 2235, Springer-Verlag, 2001, p. 17–44.

[75] J.-P. BANÂTRE, D. LE MÉTAYER. *A new computational model and its discipline of programming*, Technical report, n^o RR0566, INRIA, September 1986.

[76] J.-P. BANÂTRE, D. LE MÉTAYER. *Programming by Multiset Transformation*, in "Communications of the ACM", vol. 36, n^o 1, January 1993, p. 98–111.

[77] G. BERRY, G. BOUDOL. *The Chemical Abstract Machine*, in "Theoretical Computer Science", vol. 96, 1992, p. 217–248.

[78] D. CHEFROUR, F. ANDRÉ. *Auto-adaptation de composants ACEEL coopérants*, in "3e Conférence française sur les systèmes d'exploitation (CFSE 3)", 2003.

[79] A. J. GANESH, A.-M. KERMARREC, L. MASSOULIÉ. *Peer-to-Peer membership management for gossip-based protocols*, in "IEEE Transactions on Computers", vol. 52, n^{o} 2, February 2003, http://www.irisa.fr/paris/Biblio/Papers/Kermarrec/GanKerMas03IEEETOC.pdf.

[80] A. GEIST, A. BEGUELIN, J. DONGARRA, W. JIANG, R. MANCHEK, V. SUNDERAM. *PVM 3 Users Guide and Reference manual*, Oak Ridge National Laboratory, Oak Ridge, TN, USA, May 1994.

[81] K. GHARACHORLOO, D. LENOSKI, J. LAUDON, P. GIBBONS, A. GUPTA, J. HENESSY. *Memory Consistency and event ordering in scalable shared memory multiprocessors*, in "17th Annual Intl. Symposium on Computer Architectures (ISCA)", ACM, May 1990, p. 15–26.

[82] J. GRAY, D. SIEWIOREK. *High Availability Computer Systems*, in "IEEE Computer", September 1991.

[83] E. JEANNOT, B. KNUTSSON, M. BJORKMANN. *Adaptive Online Data Compression*, in "IEEE High Performance Distributed Computing (HPDC 11)", 2002.

[84] P. KELEHER, A. COX, W. ZWAENEPOEL. *Lazy Release Consistency for Software Distributed Shared Memory*, in "19th Intl. Symposium on Computer Architecture", May 1992, p. 13–21.

[85] P. KELEHER, D. DWARKADAS, A. COX, W. ZWAENEPOEL. *TreadMarks: Distributed Shared Memory on standard workstations and operating systems*, in "Proc. 1994 Winter Usenix Conference", January 1994, p. 115–131.

[86] L. LAMPORT. *How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs*, in "IEEE Transactions on Computers", vol. 28, n^{o} 9, September 1979, p. 690–691.

[87] P. LEE, T. ANDERSON. *Fault Tolerance: Principles and Practice*, vol. 3 of Dependable Computing and Fault-Tolerant Systems, Springer Verlag, second revised edition, 1990.

[88] F. MATTERN. *Virtual Time and Global States in Distributed Systems*, in "Proc. Int. Workshop on Parallel and Distributed Algorithms, Gers, France", North-Holland, 1989, p. 215–226.

[89] MESSAGE PASSING INTERFACE FORUM. *MPI: A Message Passing Interface Standard*, Technical report, University of Tennessee, Knoxville, TN, USA, 1994.

[90] D. S. MILOJICIC, V. KALOGERAKI, R. LUKOSE, K. NAGARAJA, J. PRUYNE, B. RICHARD, S. ROLLINS, Z. XU. *Peer-to-Peer Computing*, Submitted to Computing Surveys, Research Report, n^{o} HPL-2002-57, HP Labs, March 2002, http://www.hpl.hp.com/techreports/2002/HPL-2002-57R1.pdf.

[91] C. MORIN, R. LOTTIAUX, G. VALLÉE, P. GALLARD, D. MARGERY, J.-Y. BERTHOU, I. SCHERSON. *Kerrighed and Data Parallelism: Cluster Computing on Single System Image Operating Systems*, in "Proc. of Cluster 2004", IEEE, September 2004, http://www.irisa.fr/paris/Biblio/Papers/Morin/MorLotVal04Cluster.pdf.

[92] OMG. *CORBA Component Model V3.0*, June 2002, OMG Document formal/2002-06-65.

[93] G. PĂUN. *Computing with Membranes*, in "Journal of Computer and System Sciences", vol. 61, n^{o} 1, 2000, p. 108-143.

[94] D. RIDGE, D. BECKER, P. MERKEY, T. STERLING. *Beowulf: Harnessing the Power of Parallelism in a Pile-of-PCs*, in "IEEE Aerospace Conference", 1997.

[95] A. ROWSTRON, P. DRUSCHEL. *Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems*, in "IFIP/ACM Intl. Conf. on Distributed Systems Platforms (Middleware)", November 2001, p. 329–350.

[96] C. SZYPERSKI. *Component Software - Beyond Object-Oriented Programming*, Addison-Wesley / ACM Press, 1998.