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 1A

Table of contents

1. Team **1**
2. Overall Objectives **1**
2.1. General objectives 1
2.1.1. Parallel processing to go faster 2
2.1.2. Distributed processing to go larger 2
2.1.3. Scientific challenges of the Paris Project-Team 2
2.2. Operating system and runtime for clusters 3
2.3. Middleware systems for computational grids 4
2.4. Large-scale data management for grids 4
2.5. Advanced models for the Grid 5
3. Scientific Foundations **5**
3.1. Introduction 5
3.2. Data consistency 6
3.3. High availability 6
3.4. Localization and routing 7
3.5. High-performance communication 7
3.6. Distributed data management 8
3.7. Component model 8
4. Application Domains **8**
5. Software **9**
5.1. Kerrighed 9
5.2. PadicoTM 10
5.3. PaCO++ 11
5.4. CASPer 12
5.5. Mome 13
5.6. JuxMem 13
6. New Results **14**
6.1. Operating system and runtime for clusters 14
6.1.1. Kerrighed 14
6.1.1.1. Current achievements with Kerrighed 14
6.1.1.2. Process migration in Kerrighed 15
6.1.1.3. Thread support in Kerrighed 15
6.1.1.4. Further work on Kerrighed 15
6.1.1.5. High performance cluster-wide I/0 16
6.1.2. Checkpointing 16
6.1.2.1. Checkpointing parallel applications in clusters 16
6.1.2.2. Checkpointing parallel applications in cluster federations 16
6.1.3. GridOS 17
6.1.4. Mome and openMP 17
6.2. Middleware for computational grids 18
6.2.1. The PadicoTM framework 18
6.2.2. Parallel CORBA objects 19
6.2.3. Parallel CORBA components 19
6.2.4. Parallel component deployment for computational grids 20
6.2.5. Adaptive components 20
6.3. Large-scale data management for grids 21
6.3.1. Mome e-Toile 21

6.3.2. JuxMem 22
6.4. Advanced models for the Grid 23
7. Contracts and Grants with Industry 23
7.1. VTHD ++ 23
7.2. e-Toile 24
7.3. CASPer 24
7.4. Alcatel 25
7.5. Edf 25
7.6. Dga 25
8. Other Grants and Activities 26
8.1. Regional grants 26
8.2. National grants 26
8.2.1. ACI Globalisation des Ressources Informatiques et des Données 26
8.2.2. ACI GRID RMI 26
8.2.3. ACI GRID HydroGrid 26
8.2.4. ACI GRID DataGRAAL 27
8.2.5. ACI GRID GRID2 27
8.2.6. ACI GRID Alta 27
8.2.7. ACI GRID Core Grid 27
8.2.8. ACI GRID GRID5000 28
8.2.9. ACI Masses de Données 28
8.2.10. ACI MD GDS 28
8.2.11. ACI MD MDP2P 28
8.2.12. ACI MD GdX 28
8.2.13. Other grants 28
8.2.14. ARC RedGrid 28
8.2.15. CNRS AS 114, RTP 8 29
8.2.16. CNRS AS 115 RTP 8 29
8.3. European grants 29
8.3.1. IST POP 29
8.3.2. Core Grid 29
8.3.3. GridCoord 30
8.4. International bilateral grants 30
8.4.1. Europe 30
8.4.2. North-America 30
8.4.3. Middle-East, Asia, Oceania 31
8.5. Visits and invitations 31
9. Dissemination 32
9.1. Community animation 32
9.1.1. Leaderships, Steering Committees and community service 32
9.1.2. Editorial boards, steering and program committees 33
9.1.3. Evaluation committees, consulting 35
9.2. Academic teaching 35
9.3. Conferences, seminars, and invitations 36
9.4. Administrative responsibilities 37
9.5. Miscellaneous 37
10. Bibliography 38

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 Extension of* ENS CACHAN. *Since, the project activity is jointly supervised by a ad-hoc Committee on an annual basis. Regarding 2003, this committee met at* ENS CACHAN *on April 28.*

Head of project-team
Thierry Priol [DR INRIA]

Administrative assistant
Maryse Auffray [TR INRIA]

Staff member Inria
Gabriel Antoniu [CR]
Yvon Jégou [CR]
Christine Morin [DR]
Christian Pérez [CR]

Staff member University of Rennes 1
Jean-Pierre Banâtre [Professor]

Staff member Insa de Rennes
Jean-Louis Pazat [Assistant Professor]

Staff member Ens Cachan
Luc Bougé [Professor, ENS CACHAN Brittany Extension]

Project technical staff
Renaud Lottiaux [INRIA, DGA *COCA* Contract (since March 2003)]
David Margery [INRIA, DGA *COCA* Contract (since March 2003)]
Guillaume Mornet [INRIA, RNTL CASPER Contract]

PhD student
Alexandre Denis [MENRT Grant, ENS Lyon]
Jeremy Buisson [MENRT Grant (since October 1, 2003)]
Pascal Gallard [INRIA Grant]
Mathieu Jan [INRIA-Regional Council Grant (since October 1, 2003)]
Sébastien Lacour [INRIA Grant]
Sébastien Monnet [MENRT Grant (since October 1, 2003)]
Yann Radenac [MENRT Grant (since October 1, 2003)]
André Ribes [INRIA-Regional Council Grant]
Louis Rilling [MENRT Grant, ENS CACHAN]
Gaël Utard [INRIA Grant]
Geoffroy Vallée [Cifre EDF industrial Grant]

Long-term visitor
Ramamurthy Badrinath [IIT Kharagpur (till May 2003)]

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 allow to speed-up the design of complex manufactured products, such as cars or aircrafts, thanks to numerical simulation techniques.

As the computer performance increases rapidly, 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 simulation 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 clusters of clusters (aka *Grids*), will have to be used simultaneously to keep simulation times within reasonable bounds. Moreover, simulation will have to be performed by different research teams, each of them contributing with its own simulation code. These teams may all belong to a single company, or to different companies owning 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 four themes: *Operating System and Runtime for Clusters*, *Middleware for Computational Grids*, *Large-scale Data Management for Grids* and *Advanced Models for the Grid*. These research activities encompass both basic research, seeking conceptual advances, and applied research, to validate the proposed concepts against real applications. The project-team is also involved in setting-up a national grid computing infrastructure (*Grid 5000*) enabling large-scale experiments.

2.1.1. Parallel processing to go faster

As the performance of microprocessors, computer architectures and networks increase, a cluster of standard personal computers provides the level of performance to make numerical simulation a handy tool. This tool should be not be used only by researchers, 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, or at a coarse grain within a cluster of such PCs. This unprecedented level of performance no doubt makes numerical simulation available for a larger number of users such SMEs. It also generates new needs and demands for more accurate numerical simulation. But 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: make things faster, better and cheaper. Large scale numerical simulation will no 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 needed 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 expertize and tools such as numerical simulation codes, and even their 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

2.1.3. 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 four directions.

Operating system and runtime for clusters. The challenge is to design and build an operating system for clusters that will hide distributed resources (processors, memories, disks) to the programmers and the users. A PC cluster with such an operating system will look 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 mix both parallel and distributed processing seamlessly.

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

Advanced models for the Grid. This theme aims at contributing to study unconventional approaches for the programming of grids based on the chemical metaphors.

2.2. Operating system and runtime for clusters

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

We believe that cluster programming is still difficult as 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 rather than the vision of a set of independent computers, each with resources locally managed.

Several attempts to build an SSI have been made at the middleware level as Beowulf [87], PVM [74] or MPI [90]. However, these environments provide only a partial SSI. Our approach in 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) and kernel resources (process, memory pages, data streams, files) 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 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 application whatever the communication paradigm they are based on. Even, if backward error recovery (BER) has extensively been studied from the theoretical point of view, it is still challenging to efficiently implement BER protocols transparently to the applications. There are very few implementations of recovery 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 different 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 from the target machine and its operating system. The main originality of our approach is to use the concept of distributed shared memory as the basic communication mechanism within the runtime. We are essentially interested in Fortran and its OpenMP extensions [85]. 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 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 focus a lot on resources management like discovery, registration, security, scheduling, etc. However, they provides very few support for grid-oriented programming model.

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 pertinent solution. However, they need to be tailored for scientific applications, in particular with respect of the encapsulation of parallel codes into objects or components, the communications between "parallel" objects or components, the required runtime support, the deployment and the 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 code, which should take into account the parallelism of resources. It is thus required to efficiently combine both world into a coherent one.

The second issue concerns the simplicity and the scalability of communications 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 control flow should be used not to congest networks. A crucial aspect of this issue is the support for data redistribution which should be involved in the communication between parallel codes.

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 *simultaneously* active within a same process. Third, solutions to the two previous issues should support high performance constraints to be accepted by users.

The deployment of applications is another issue. Not only, it is important to constrain the deployment by specifying the requirement in term of the computational resource (Gflop/s, amount of memory, etc.), but it is also crucial to specify the constraints related to communication resources such as the amount of bandwidth or the latency between computational resources.

The last issue deals with the dynamic nature of computational grids. As targeted applications may last for very long time, grid environment is expected to change. Not only middleware systems should support adaptability but they should be able to detect variations and should be able to self-adapt. For example, an application may be partially redeployed to take benefit of 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 computation. 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 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, a few constraints need to be addressed. The clusters which make up the grid are not guaranteed to remain constantly available. 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.

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 large scales and can serve as inspiration source.

Finally, in grid applications, data are generally shared 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 models for the Grid

Till now, research activities related to the Grid have been focused on the design and implementation of middleware and tools to experiment grid infrastructure with applications. Very few attention has been paid to programming models suitable for such widely computing infrastructures. Programming of such infrastructures is still 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. In particular, we are considering unconventional approach based on the *chemical reaction* paradigm, and more precisely the Gamma Model [3].

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, 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.

3. Scientific Foundations

3.1. Introduction

Research activities within the PARIS Project-Team encompass 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 physical access only to its local memory. Implementation 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 written operation performed by another processor previously. Within a distributed or parallel a system, the notion of the *last* memory access is sometimes undefined since there is no global clock that gives 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* [80] 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 appear 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 run parallel programs correctly, 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* [75] has been thoroughly studied since it is well adapted to programming parallel scientific applications. The principle behind Release Consistency that memory accesses are (should?) always be guarded by synchronization operations (locks, barriers, etc.), so that the shared memory system only needs to be consistent 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 have been proposed of Release Consistency [78]: 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 two alternative implementations differ in the number of messages that needs to be sent/received, and in the complexity of the implementation [79]. Implementations of Release Consistency rely on the use of a logical clock such as a vector clock [82]. 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 practically impossible. 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* [76] 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* [81] based on redundancy can be implemented.

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 is 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 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 a 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. Localization and routing

Recent research on emerging *peer-to-peer* (P2P) systems [86] has focused on designing adequate localization and routing strategies for large-scale, highly-decentralized environments. The proposed algorithms have properties that address the main requirements of such environments: high scalability, fault tolerance (with respect to node or link failures), no (or very little) dependence on centralized entities. The very first approaches (illustrated by *Napster*) still use a centralized directory for data localization, then switch to direct P2P interaction for the actual data transfers. Later, fully distributed, flooding-based approaches (e.g., *Gnutella*) have been proposed. A second generation of P2P systems (e.g., *KaZaA*) have combined the previous techniques by integrating the notion of super-peer: localization is flooding-based between the super-peers, which serve as local directories for groups of regular peers. However, flooding strategies have one main weakness: since they generate a lot of traffic, a limit has to be set on the number of times queries are re-propagated. As a result, queries for data may fail, whereas the data are actually stored in the system.

In order to provide both high fault tolerance and the guarantee to always reach data available in the network, recent research has focused on localization schemes based on *Distributed Hash Tables* (DHT). This promising approach is illustrated by *Chord* (MIT), *Pastry* (Microsoft Research) and *Tapestry* (UC Berkeley) and has also been used for the latest major version (2.0) of the *JXTA* generic environment for P2P services started by Sun Microsystems. Efforts are currently under way, in order to define a common API for such DHT-based systems [72].

3.5. High-performance communication

High-performance communication [71] is uttermost crucial for parallel computing. However, it is less important in distributed computing, whereas interoperability is more important. The quest for high-performance communication has led to the development of specific hardware technologies along the years: *SCI*, *Myrinet-1*, *VIA*, *Myrinet-2000*, *InfiniBand*, etc. A dedicated low-level communication library is often required to fully benefit from the hardware specific feature: *GM* or *BIP* for *Myrinet*, *SISCI* for *SCI*, etc. To face the diversity of low level communication libraries, research has focused on generic high-performance environments such as *Active Message* (Univ. of Berkeley), *Fast Message* (Univ. of Illinois), MADELEINE (LaBRI, Bordeaux), *Panda/Ibis* (Univ. of Amsterdam) and *Nexus* (Globus Toolkit). Such generic environments are usually *not* assumed to be directly used by a programmer. Higher-level communication environment are specifically designed: *PVM*, MPI or software DSM such as *TreadMarks* are such examples in the field of parallel computing. While high performance communication research has mainly focused on system-area networks, the emergence of grid computing enlarges its focus to wide-area networks and, more specifically, to *high-bandwidth, wide-area networks*. Research is needed to efficiently utilize such networks. Some examples are adaptive dynamic compression algorithms, and especially parallel stream communication.

Previous work [70] has shown that high-performance communication not only require an adequate communication library, but also demand some cooperation with the *thread scheduler*. It is particular important as more and more middleware systems as well as applications are multithreaded. Another related issue, which devotes further research, is to minimize network reactivity without generating too much overhead.

3.6. Distributed data management

Past research on distributed data management led to 3 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* [73] platform provides data access mechanisms (*Globus Access to Secondary Storage*), 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 and use them as temporary buffers for efficient data transfers across a wide-area network. Transfer management is still at the user's charge. Besides, IBP does not handle dynamic join/departure of storage nodes and provides no consistency guarantee for multiple copies of the same data.

In contrast, *Distributed Shared Memory* (DSM) systems provide transparent data sharing, via a unique address space accessible to physically distributed machines. Within this context, a variety of consistency models and protocols have been defined. These systems do offer transparent access to data: all nodes can read and write data in a uniform way, using a unique identifier or a virtual address. It is the responsibility of the DSM system to localize, transfer, replicate data, and guarantee their consistency according to some semantics. Nevertheless, existing DSM systems have generally shown satisfactory efficiency only on small-scale configurations, typically, a few tens of nodes.

Recently, *peer-to-peer* (P2P) has proven to be an efficient approach for large-scale data sharing [83]. The peer-to-peer model is complementary to the client-server model: the relations between machines are symmetrical, each node can be client in a transaction and server in another. This paradigm has been made popular by *Napster*, *Gnutella* and *KaZaA*. Such systems have proven able to manage very large configurations (millions of nodes) with a very high volatility. However, we can note that most P2P systems focus on sharing immutable files: the shared data are *read-only* and can be replicated at ease. Recently, some mechanisms for sharing *mutable* data in a P2P environment have been proposed by systems like OceanStore and Ivy, with restricted use (no multiple writers nor conflict resolution).

3.7. Component model

Software component technology [89] 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 [84], which is part of the latest CORBA [84] 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 CCA Forum [69] 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.

4. Application Domains

Key words: *Scientific computing*, *coupling of numerical codes*.

The project-team research activities address in priority 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 and 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.

If scientific computing is our primary target to apply the results gained by the project-team, we do not exclude other kind of applications such as multimedia or discrete-event distributed applications for which our research can be applied.

5. Software

5.1. Kerrighed

Participants: Pascal Gallard, Renaud Lottiaux, David Margery, Christine Morin, Louis Rilling, Gaël Utard, Geoffroy Vallée.

Key words: *Cluster operating system*, *single system image*, *global and dynamic resource management*, *distributed shared memory*, *process migration*, *global scheduling*, *cooperative caching*, *remote paging*, *high availability*, *backward error recovery*.

Contact: Christine Morin

URL: http://www.kerrighed.org/

Status: Registered at APP, under Ref. IDDN.FR.001.480003.005.S.A.2000.000.10600

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

Presentation: KERRIGHED (formerly known as *Gobelins*) 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 all the SSI properties. Global resource management enables transparent distribution of resources throughout the cluster nodes and 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 to have to restart applications from the beginning when node failure happens.

To avoid mechanism redundancy and conflicting decisions in different distributed resource management services and to decrease the software complexity of such services, KERRIGHED resource management services are built in an unified and integrated way.

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 may be executed without modification on top of KERRIGHED and further optimized if needed.

KERRIGHED is not an entirely new operating system developed from scratch. 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 of 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`) is available to support users of KERRIGHED.

Current status: KERRIGHED includes 80,000 lines of code (mostly in C). It represents 140 person-months of effort. The development of KERRIGHED started in late 1999. The stable release of KERRIGHED is Version V0.72 [51] (October 2003). It provides a complete *Pthread* support, allowing to execute legacy OpenMP and multithreaded applications on a cluster without any recompilation.
KERRIGHED currently includes 6 Linux modules and a limited patch to Linux Kernel 2.2.13. A port to Linux Kernel 2.4.20 is currently in progress, and a new version of KERRIGHED will be released in early December.
Several demonstrations of KERRIGHED have been presented this year at *Linux Expo* (Paris, February 2003, Louis Rilling and Christine Morin), the *IPDPS Conference* (Nice, April 2003, Geoffroy Vallée and Louis Rilling), EDF *R&D Printemps de la recherche* (Clamart, May 2003, Renaud Lottiaux) and *Euro-Par Conference* (Klagenfurt, Austria, August 2003, Pascal Gallard and Gaël Utard) [32].
KERRIGHED is currently experimented by *Cap Gemini Ernst and Young*, *ONERA CERNT*, DGA *CELAR* in the framework of COCA contract, as well as by EDF. More than 100 external downloads of KERRIGHED have been recorded since November 2002.

5.2. PadicoTM

Participants: Alexandre Denis, Christian Pérez, Thierry Priol, Benoît Hubert.

Key words: *Grid*, *middleware system*, *communication framework*.

Contact: Christian Pérez

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

Status: Registered at APP, under Ref. 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 runtimes. 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 cooperative 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, a very useful one aims at hiding the differences between different CORBA implementation. The first version was called *Ugo* (available in the 0.1.x Series). It has since been replaced by *myCORBA* in the next version.
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 operation 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 local file system as well as through a network thanks to a SOAP daemon being part of the service.
A public mailing list (`padico-users@listes.irisa.fr`) is available to support users of PADICOTM.

Current status: The development of PADICOTM has started end of 2000. It represents XXX person-month effort.
The stable release of PADICOTM is Version 0.1.5 (November 2002). The unstable version (CVS version) is 0.2.0beta3.
The stable version (0.1.x series) includes the PADICOTM core, PadicoControl, Ugo and external software: a PADICOTM-enabled version of *omniORB* (3.0.2), a PADICOTM-enabled version of *MPICH* (1.1.2), a customized version of PM^2, and a regular version of *Expat* (1.95.2)
PADICOTM 0.1.5 (without external software) includes 31,000 lines of C and C++ (ca. 900 kB), 2,300 lines of JAVA (ca. 70 kB) and 7,000 lines of`shell`, `make` and `configure` scripts (ca. 200 kB).
The CVS version (0.2.x series) includes an updated version of PADICOTM core (bug fixes as well as some internal rewriting), *PadicoControl*, *myCORBA* (replaces *Ugo*) 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 supported middleware systems. Current supported middleware systems are *omniORB3*, *omniORB4* and *Mico* 2.3.x for CORBA, *MPICH* 1.1.2 and *MPICH* 1.2.4 for MPI and *gSOAP* 2.2.x for SOAP.

Users: 90 external downloads whose 63 unique IPs between July 2002 and November 2003.
PADICOTM has been funded by the French ACI GRID RMI. As we are aware of, it is currently used by several French projects: ACI GRID HydroGrid, ACI GRID EPSN. PADICOTM has been demonstrated at the *SuperComputing 2002* Conference, in Baltimore, MD, USA.

5.3. PaCO++

Participants: André Ribes, Christian Pérez, Thierry Priol.

Key words: *Grid*, *middleware system*, *CORBA*, *data parallelism*.

Contact: Christian Pérez

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

Status: Prototype under development.

License: Yet to be decided.

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 accessible externally through an object reference whose interpretation is identical to a standard CORBA object.
PACO++ extends CORBA but not to modify the model because we aim at defining a *portable* extension to CORBA so that it can be added to any CORBA implementation. This choice stems also from the consideration that the parallelism of an object appears to be an implementation issue of the object. Thus, the OMG IDL is not required to be modified.
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: PACO++ is still in a development phase. A public version should be released during the winter 2003-2004. PACO++ has been successfully tested on top of three CORBA implementations: *Mico*, *omniORB3* and *omniORB4*. Moreover, it supports PADICOTM.
The current version of PACO++ includes 7,600 lines of JAVA (ca. 252 kB), 6,900 lines of Python (ca. 390 kB), 13,400 lines of C++ (ca. 390 kB) and 2,200 lines of `shell`, `make` and `configure` scripts (66 kB).
PACO++ is supported by the French ACI GRID RMI. Non-public beta versions are currently used by several French projects: ACI GRID HydroGrid, ACI GRID EPSN and RNTL VTHD ++.

5.4. CASPer

Participants: Guillaume Mornet, Jean-Louis Pazat.

Key words: *Application Service Provider*, *Grid Services*.

Contact: Jean-Louis Pazat, http://www.telecom.gouv.fr/rntl/AAP2001/Fiches_Resume/CASPER.htm

License: LGPL

Presentation: CASPER aims at providing an *Application Service Provider* (ASP) for Grid computing.
The server side is based on the *Globus Toolkit* (GTK 3): CASPER is made of services that communicate with well-defined protocols, mainly XML-RPC calls (for *Grid Services*), JDBC connections (for databases) and HTTP connections. The ASP manages authentication, user interface, persistent data storage, job scheduling. Batch queues provide computing power for jobs submitted by users through the ASP.
On the Client side, CASPER can work with any standard Web Browser, this ensures that CASPER will be usable from most platforms.
CASPER is partly built using *components off the shelf* (COTS) for the Web browser, Web server (*TOMCAT*), SQL database (*MySQL*). The job managers currently targeted are OpenPBS, LSF and LoadLeveler. A Distributed Job Manager (*XtremWeb*) will be also be integrated within CASPER as a special job manager.
Security in CASPER is managed at different layers: first, we secure the HTTP connection between the client and the ASP (SSL and certificates), then we secure the communications between the ASP and batch queues (services have certificates). This is needed because batch queues may spread across Virtual Organizations).
CASPER provides a Job Scheduler as a service responsible for scheduling job requests from users to the appropriate batch queue. Criterion for scheduling include: the required architecture, the list of queues the user is authorized to run jobs on, the current state of the queue, the job type (e.g., parallel or distributed), etc.
User management is done by a module that takes care of generating certificates, and updating access control lists (ACLs).
A CASPER application is made of a GUI which main functions are selecting job submission parameters, and a Job Runner that requests a job submission to the job scheduler in order to submit the code that executes the simulation.
Computations results are transferred from the batch queue to the ASP using the RFT Grid Service (which relies on a secured FTP protocol). These files will be stored on the ASP, with owner information. The result files can be remotely viewed (if a suitable viewer applet is available), downloaded, or deleted. The CASPER security manager controls access to the files.

Current status: The CASPER ASP is under development. The current release is for internal testing only. This project started in October 2003 and is supported by a RNTL contract. The main industrial contractor is EADS-CCR.

5.5. Mome

Participant: Yvon Jégou.

Key words: *DSM*, *data repository*.

Contact: Yvon Jégou

Status: Prototype under development

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

License: APP registration in the future, license type not yet defined (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 DSM handles the consistency of mapped memory regions at the page-level. Two consistency models are currently implemented and can be selected by the user programs at the page level: the classical sequential model and an explicit weak model. MOME initial target was the execution of programs from the high performance community which exploit loop-level parallelism using a SPMD computation model. the current release of MOME supports

- page aliasing: the same page can be mapped twice (or more) in the same address space. This feature is used in the implementation of a DSM-based coupling library.
- heterogeneous applications: different programs (binaries) can share data (be coupled) through the same instance of the DSM.
- checkpointing: on checkpointing request from the application, the DSM guarantees that two copies of each DSM page are present on two different nodes. In case of failure, it is possible to restart the DSM and to recover all the pages of the last checkpoint as long as one node maximum that participated to the checkpoint has crashed.
- dynamic connection of processes: MOME can be started as a background daemon and supports the dynamic connection of processes. This possibility allows the implementation of persistent data repositories.

The current developments around MOME involve the implementation of an OpenMP runtime system, the implementation of a DSM-based coupling library and the development of a persistent data repository for the grid.

Current status: MOME is implemented in C (50,000 lines) and represents a 24-person-month effort. The current release is MOME 0.8. The DSM is used in ALCATEL collaboration (checkpointing), in VTHD contracts (code coupling using a DSM), in *e-Toile* project (DSM-based data-repository for grid computing) and in the POP project (OpenMP runtime).

5.6. JuxMem

Participants: Gabriel Antoniu, Luc Bougé, Mathieu Jan.

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

Contact: Gabriel Antoniu, http://www.irisa.fr/paris/Juxmem/welcome.htm

License: Not yet defined.

Presentation: JUXMEM is an experimental platform for building a data-sharing service for grid computing. The 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. Several studies on replication strategies for fault tolerance and consistency protocols for volatile environments are under way within the framework provided by JUXMEM. A more detailed description of the approach followed by JUXMEM is given in 6.3.2.

Current status: Implemented in JAVA, based on the *JXTA* generic platform for P2P services. 5,000 lines of code. Implementation started in February 2003. JUXMEM is the starting point of a Grid Data Service (GDS) that will be built in collaboration with the ReMaP/GRAAL (Lyon) and REGAL (Paris) research groups, within the framework of the GDS project of the ACI MD (see Section 8.2.9).

6. New Results

6.1. Operating system and runtime for clusters

Key words: *Cluster*, *cluster federation*, *operating system*, *distributed system*, *single system image*, *global scheduling*, *process migration*, *multithreading*, *high performance communication*, *data stream migration*, *distributed shared memory*, *cooperative caching*, *remote paging*, *checkpointing*, *high availability*, *Pthread*, *OpenMP*, *MPI*, *peer-to-peer*, *self-organizing system*, *synchronization*.

6.1.1. Kerrighed

Participants: Pascal Gallard, Renaud Lottiaux, Christine Morin, Louis Rilling, Gaël Utard, Geoffroy Vallée.

Clusters are not only the most widely used general high-performance computing platforms for scientific computing, but they have also become the most dominant platform for high-performance computing today, according to the http://www.top500.org/ site. While the cluster architecture is attractive with respect to price/performance, there still exists a great potential for efficiency improvements at the software level. System software requires improvements to better exploit the cluster hardware resources and to ease cluster programming.

Since 1999, the PARIS Project-Team is engaged in the design and development of KERRIGHED, a genuine *Single System Image* cluster operating system for general high-performance computing [15][33] (Kerrighed is a registered trademark). 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. An SSI should offer four properties: (1) *Resource distribution transparency*, i.e., offering processes transparent access to all resources, and resource sharing between processes whatever the resource and process location; (2) *High performance*; (3) *High availability*, i.e., tolerating node failures and allowing application checkpoint and restart; and (4) *Scalability*, i.e., dynamic system reconfiguration, node addition and eviction, transparently to applications.

6.1.1.1. Current achievements with Kerrighed

In 2003, thanks to the recruitment of two expert engineers within the COCA Contract, we have integrated the results obtained last year by several PhD students into a unique prototype. Two major releases (KERRIGHED V0.70 [50] and V0.72 [51]) have been delivered. The robustness of KERRIGHED has been significantly enhanced, and several new functionalities have been implemented. KERRIGHED V0.72 is now suitable for the execution of applications provided by our industrial partners (EDF, DGA).

In 2003, we have focused on the implementation of a *configurable* global scheduler for KERRIGHED [37][17]. It should be able to adapt the global scheduling policy to the workload characteristics, and to change it dynamically depending on the cluster load. All components of the KERRIGHED scheduler can be hot-plugged

or hot-stopped. A development framework allowing to easily implement global scheduling policies has been designed and implemented. These schedulers may rely on new components, developed without any kernel modification or on components existing in KERRIGHED. Some preliminary global scheduling policies have been experimented [38].

In the near future, we plan to implement a number of global scheduling policies in KERRIGHED and to experiment them with respect to workloads made up of real sequential and parallel applications provided by EDF. We will also design and implement a simple batch system exploiting KERRIGHED features to meet users requirements.

6.1.1.2. Process migration in Kerrighed

In order to cope with the migration of communicating processes, we have designed the *Kernet* System [29]. It supports the global management of any data stream (*socket*, *pipe*, *char* devices) in KERRIGHED. In 2003, *unix* et *inet* sockets have been implemented on top of *Kernet*. Other data stream interfaces will be implemented in 2004. Performance evaluations have been carried out with MPI applications based on the MPICH environment. (Note that no modification of the MPICH environment is needed to execute it within KERRIGHED.) They demonstrate that a MPI process can be migrated in KERRIGHED without any performance degradation incurred by communications taking place after migration [13].

Kernet relies on the*Gimli*/*Gloïn* System. It is a portable, high-performance communication system, providing a kernel-level send/receive interface to KERRIGHED distributed services. We have revisited the design of *Gimli*/*Gloïn* to obtain better performance, and to extend its interface with active messages and pack/unpack primitives. The implementation of the new *Gimli*/*Gloïn* architecture, which offers high-performance communication both at kernel level and at user level, is in progress. In addition, we plan to implement a new *Gloïn* device to better exploit the Myrinet technology.

6.1.1.3. Thread support in Kerrighed

Today, KERRIGHED offers a complete support of the *Posix* thread standard. This important result has been obtained thanks to our previous results on distributed shared memory. It also crucially relies on the work carried out in 2003 on the design and implementation of distributed mechanisms for proper thread termination, cluster-wide signal management, and distributed synchronization facilities (locks, barriers, etc.) compatible with preemptive thread migration. KERRIGHED *Pthread* Interface has been validated by executing existing OpenMP applications compiled with the unmodified *Omni* 1.4 OpenMP compiler targeting *pthread* in SMP multiprocessors [31]. The correct execution of the 288 tests provided the *Omni* compiler to ensure its correct installation on a given architecture demonstrates that KERRIGHED is now mature to support multithreading on a cluster. The OpenMP version of the *HRM1D* EDF application has also been successfully executed on Kerrighed. It includes 7,000 lines of Fortran code. However, performance has to be improved. In the future, we plan to explore cluster-aware compilation methods, that produce more efficient code for OpenMP programs. They should also provide OpenMP developers with tools to better understand the performance bottlenecks of their applications so that they can tune their parallelized algorithms.

6.1.1.4. Further work on Kerrighed

Future work on KERRIGHED is three-fold. First, we will continue the development of KERRIGHED with the design of a distributed file system based on containers [65]. We will also integrate checkpointing mechanisms for parallel applications. KERRIGHED will be ported to a 64-bit architecture based on Opteron processors. Second, we will work further on high-availability issues. On the one hand, we will continue the work started this summer in cooperation with Rutgers University during the internship of Pascal Gallard. It is devoted to high-availability issues in exploiting the read and write RDMA features provided in the last generation of Myrinet adapters. On the other hand, we will work on dynamic reconfiguration mechanisms for KERRIGHED distributed services. Last, we will pursue our efforts to extend the community of KERRIGHED users. In cooperation with EDF, we plan to build an OSCAR package based on KERRIGHED (SSI-OSCAR) during the post-doctoral internship of Geoffroy Vallée at Oak Ridge National Laboratory.

6.1.1.5. High performance cluster-wide I/0

High performance I/O are of primary importance for the applications executed on clusters. Some applications, like numerical computation or VOD, demand high-bandwidth sequential accesses. Other, like mail or Web servers, benefit from low-latency data and meta-data operations. As of today, no cluster file system provides performance for the whole range of access patterns.

Rather than putting forward another middleware, we explore a new approach to make the operating system capable of *efficient distributed I/O*. We propose to manage a cluster-wide cache, consisting of both data and meta-data, through Distributed Shared Memory (DSM) techniques [36]. Our goal is to shorten the data path, and efficiently overlap I/O, communication and computation, in order to avoid dedicated I/O nodes and network-attached storage. With the I/O system we propose, it is possible to take advantage of Direct Remote Access (DMA) transfers, thus avoiding stressing the local bus of a node. Our system is also compatible with striping and mirroring as in software RAID systems.

A first prototype has been implemented based on a modified version of the Linux Kernel. We plan to complete this prototype to validate our approach with respect to standard benchmarks. We also plan to provide an MPI-IO interface.

6.1.2. Checkpointing

Participants: Ramamurthy Badrinath, Christine Morin, Louis Rilling, Geoffroy Vallée, Sébastien Monnet.

6.1.2.1. Checkpointing parallel applications in clusters

Backward error recovery involving checkpointing and restart of tasks is an important component of any system providing fault tolerance to applications distributed over a network. In KERRIGHED, one of our objectives is to be able to checkpoint and restart any kind of scientific application: they may range from sequential applications to parallel applications, with communication based on message passing shared memory, or even both of them.

We have identified common mechanisms for implementing a wide variety of checkpointing and rollback recovery protocols for both message passing and distributed shared memory systems [23]. The idea is to treat pages as separate entities similar to regular tasks, and provide a mechanism to track *direct dependencies* among tasks and memory pages. This mechanism is thus common to both distributed shared memory and message-passing systems. It can moreover be efficiently implemented, since the overhead of each interaction is very light, both in terms of computation and control information. The proposed mechanism can finally support several optimizations discussed in the literature.

We have carried out a first implementation of a *coordinated checkpointing protocol* within KERRIGHED cluster operating system for multithreaded applications [24]. Checkpointing the private state of a thread involves the same basic mechanisms as those used for process migration. Additionally, a mechanism to checkpoint container pages *incrementally* has been implemented. As the checkpoint storage has a high impact on performance during fault-free executions, KERRIGHED can save checkpoints either in the memory of two nodes or into disks. Synchronization is an important issue when designing a checkpoint/recovery protocol for shared memory applications. In fact, parallel applications traditionally use locks and barriers, that may incur causality dependences between processes. We have studied how to extend our previous work on dependence tracking to deal with synchronization [43]. Both locks and barriers introduce resource contention conflicts at recovery time, which need to be resolved by heuristics. Moreover, due to these conflicts, a unique latest recovery line is not defined, which requires using additional heuristics.

This work will be continued next year. We will finalize the implementation in KERRIGHED of all the mechanisms needed to checkpoint and recover parallel applications communicating by message (for instance, MPI applications) or shared memory (for instance, OpenMP applications). A preliminary global coordinated checkpointing strategy will be evaluated. Comparison with other systems will be carried out in the framework of the *Procope* 2004 bilateral collaboration with the University of Ulm, Germany.

6.1.2.2. Checkpointing parallel applications in cluster federations

Federations of clusters (aka *clusters of clusters*) are very useful for applications like large-scale code coupling. Faults may appear very frequently, so that checkpointing strategies should definitely be provided to restart

the applications in the event of a node failure in a cluster. To take into account the constraints introduced by clusters federation architecture, we propose a *hierarchical checkpointing protocol* [53]. It uses a regular synchronous approach inside individual clusters, but only quasi-synchronous methods between clusters. Our protocol has been evaluated by simulation. It fits well for applications that can be divided into modules with little inter-module communication.

6.1.3. GridOS

Participants: Christine Morin, Louis Rilling.

Large companies exploit several medium-size clusters distributed on several geographic sites. Some applications, such as those using code coupling, may overcome the capacities of one single cluster. A solution is to run each component of the code on a different cluster. Moreover, for a given cluster, a limited amount of applications can be run at the same time. However, other clusters in the same company may be idle or underloaded at the same time, and hence could be enrolled in the computation. What is needed here is a *Grid-aware* operating system, that could federate clusters in order to make them cooperate, in particular for sharing resources.

We have worked on the design of such a *Grid-aware* operating system. It should be able to manage a large number of nodes, and to deal with the dynamicity inherent to a federation, where multiple reconfigurations (node connection, disconnection, or failure) may be in progress at the same time. Our proposal is based on a *peer-to-peer* infrastructure. The idea is to build a *virtual overlay network* for a federation. Such a network provides a key-based routing protocol, making transparent the physical location of any object named by a key. The *Grid-aware* operating system would encompass several distributed services such as, for instance, services assembling a federation, managing and scheduling applications, controlling resource access, managing a virtual shared memory and a distributed file system, etc.

This year, we have implemented (using C) the peer-to-peer overlay network inspired from *Pastry* [88]. It will serve as the basis for implementing the distributed services at the federation level.

The first service that we have studied is a service for executing distributed applications using the shared memory paradigm in a cluster federation. This raises the problem of executing shared memory parallel applications on dynamic and large-scale systems. The shared memory is private to each application, it is volatile, and the application components transparently access shared memory objects via their usual address space. The peer-to-peer system tolerates up to simultaneous reconfiguration events (node failure, disconnection, or join) and an infinite number of reconfigurations. We have designed a coherence protocol similar to Kai Li's protocols for replicas of memory objects [56]. The protocol uses the peer-to-peer architecture to handle the simultaneous reconfiguration events. The number of simultaneous such events which can be handled is a parameter of the system. However, an infinite number of reconfigurations can be supported with a fail-stop model. Failures are tolerated using backward error recovery and replicated automata. This avoids restarting the applications when possible. We have proved that the protocol preserves the coherence of replicated memory objects, despite of simultaneous reconfiguration events, and guarantees liveness if communications are reliable.

Optimizations to this protocol will be studied in the near future and both theoretical and experimental evaluations will be performed. Furthermore, we plan to study a home-based coherence protocol implementing a release consistency memory model.

6.1.4. Mome and openMP

Participants: Yvon Jégou, Christian Pérez.

MOME initial target was the execution of programs from the high performance community which exploit loop-level parallelism using a SPMD execution model. This implementation makes a clear distinction between shared and private data. The allocation of the shared data in the shared space must be explicitly requested by the application. This execution model is consistent with the HPF language where the variables are implicitly private and the shared variables must be explicitly specified.

The OpenMP specification targets SMP architectures: shared memory multiprocessors. In the OpenMP model, all variables are implicitly shared. The private variables (one instance per thread) must be explicitly

specified. It is not possible through static analysis to decide at compile-time which objects are shared and which ones are private.

The MOME DSM implementation and the associated runtime system have been adapted in order to support standard OpenMP codes without adding complexity to compilers: the thread stacks can now be allocated in the shared space, the signal handlers are executed on private stacks, the DSM internal code never read or write in the application space, the distributed synchronization objects are allocated in the shared space but the primitives do not touch the objects etc.

A new implementation of the `nth_lib` runtime system from the IST POP project on the MOME DSM is under progress and the experimentations will start in the near future. The integration of the release consistency model in MOME is planned.

6.2. Middleware for computational grids

6.2.1. The PadicoTM framework

Participants: Alexandre Denis, Christian Pérez, Thierry Priol, André Ribes, Benoît Hubert.

Key words: *Communication framework*, *CORBA*, *MPI*.

Computational grids differ from previous computing infrastructure as they exhibit parallel and distributed aspects: a computational grid is a set of various and widely *distributed* computing resources, which are often *parallel*. Therefore, a grid usually contains various networking technologies — from system area network through wide area network.

PADICOTM is a communication framework that decouples application middleware systems from the actual networking environment. Hence, applications become able to transparently and efficiently utilize any kind of communication middleware (either parallel or distributed-based) on any network that they are deployed on. Moreover, to support advanced grid programming models, PADICOTM is able to concurrently support several communication middleware systems.

PADICOTM achieves these functionalities by implementing a *dual-abstraction* model which is organized in 3 layers: arbitration, abstraction, and personalities. The two paradigms, parallel and distributed, are present at each level. Therefore, cross-paradigm translation is performed only when required (i.e. distributed middleware atop parallel hardware or parallel middleware atop distributed networks) with no bottleneck of features. The lowest level layer is the arbitration layer. It goal is to provide arbitrated interfaces, i.e. a consistent, reentrant and multiplexed access to every networking resources, each resource is utilized with the most appropriate driver and method. On top of the arbitration layer, an abstraction layer aims at providing abstract interfaces well suited for their use by various middleware systems, independently from the hardware. The abstract layer should be fully transparent: the interfaces are the same whatever the underlying network is. The last layer is a *personality* layer which is able to supply various standard APIs on top of the abstract interfaces. Personalities are thin wrappers which adapt a generic API to make it look like another API. They do no protocol adaptation nor paradigm translation; they only adapt the syntax.

During the year 2003, we have finalized the specification of the three-layer dual-abstraction model and have modified PADICOTM accordingly. The arbitration layer in PADICOTM is called *NetAccess*, which contains two subsystems: *SysIO* for access to system I/O (sockets, files), and *MadIO*, based on MADELEINE, for multiplexed access to high-performance networks. A *core* handles a consistent interleaving among the concurrent polling loops. *NetAccess* is open enough so as to allow the integration of other subsystems beside *MadIO* and *SysIO* for other paradigms such as *Shmem* SMP nodes for example. The two abstract interfaces in PADICOTM are *VLink* for distributed computing, and *Circuit* for parallelism. The *VLink* interface has been implemented on top of several drivers: *MadIO*, *SysIO*, Parallel Streams, AdOC (a dynamic adaptive compression library) and *loopback*. The *Circuit* interface, which manages communications on a definite set of nodes, has been implemented on top of *MadIO*, *SysIO*, *loopback* and *VLink*. As a given instance of *Circuit* can use different adapters for different links, it is possible to build a circuit using different kinds of communication.

6.2.2. Parallel CORBA objects

Participants: Christian Pérez, Thierry Priol, André Ribes.

Key words: *Grid*, *distributed object*, *parallelism*, *CORBA*.

The concept of (distributed) parallel object appears to be a key technology for programming (distributed) numerical simulation. It joins the well known object oriented model with a parallel execution model. Hence, a data distributed across a parallel object 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 points. It targets a portable extension to CORBA so that it can be added to any implementation of CORBA. It advocates the parallelism of an object is mainly an implementation issue: it should not be visible to users but in some special occasions. Hence, the OMG IDL is no longer modified.

In 2002, the development of PACO++ was started to validate the *portable* parallel CORBA object concept. A first implementation was done which was validated with an EADS application that manipulated block-cyclic distributed matrices of complex number. It was composed of an IDL-toIDL compiler and a (C++) runtime part. Both tools only required a compliant CORBA implementation. Moreover, the PACO++ runtime handled threads, intra-parallel object communications and data distributions thanks to abstract interfaces.

Continuing the development effort, we have worked on stabilizing the code of PACO++ so that to be able to deliver it to our partners in the HydroGrid ACI GRID project and to the ReMaP/GRAAL research team (LIP, Lyon, France). We plan to release a public version during the winter 2003–2004.

The data distributed abstraction of PACO++ was quite primitive. Thanks to the support of the INRIA ARC *RedGrid*, we are working on refining the role of the PACO++ runtime. An important motivation was to be able to integrate a communication scheduling functionality to take into account the capabilities of the underlying networks. We succeed in integrating the scheduling library done by the Algorille research team (Nancy, France) into an experimental prototype. Its communication performance does not decrease when the number of sender increases contrary to other prototypes that face network congestion when the number of sender becomes too large with respect to the WAN bandwidth.

Another issue with parallel object we have started to study concerns the management of exception. The problem is to define the semantic of an exception raised during the invocation of a parallel operation invocation. We have identified various scenarios (a single exception, several identical exceptions, several different exceptions, etc) and the semantic that can be associated to them (standard exception mechanism, group of exceptions, priority of exceptions, etc). It is still on going work.

Future work can be divided into three parts. First, we will continue the development of PACO++ with the integration of a support for communication scheduling as well as a more open model for distributed data. Second, the work on parallel exceptions will be finished and implemented into PACO++. Last, we would like to show the concept of parallel object can be applied to other technologies than CORBA, like for example Web Services or Peer-to-Peer.

6.2.3. Parallel CORBA components

Participants: Christian Pérez, Thierry Priol, André Ribes.

Key words: *Grid*, *software component*, *parallelism*, *CORBA Component Model (CCM)*.

Software component technology represents the next attempt to deal with the complexity of software programming. CORBA component model (CCM) is the OMG standard for component-based distributed programming. Like CORBA objects, CORBA components suffer limitation with respect to parallelism. Our goal being to study the concept of parallel component, CCM appears to be a reasonable technological choice as it specifies the whole life cycle of a component, including its packaging and its deployment.

We have proposed to define a parallel component as a collection of identical sequential components that executes all or some parts of its services in parallel. This definition allows us to apply the experience acquired with PACO++. The OMG IDL3, which is the component abstract view, does not need to be modified. The

parallelism, specified into an auxiliary file, can be attached to the implementation definition language (CIDL). Hence, it should be possible to implement parallel components as an extension of CCM implementations.

To evaluate the pertinence of our parallel component definition, we have implemented two prototypes of parallel CORBA components based on two preliminary CCM implementations: OpenCCM, a JAVA implementation, and MicoCCM, a C++ implementation. For both prototypes, the definition of parallel component was pertinent as no particular difficulty was found. Moreover, both prototypes perform as expected. The latency measurements do not show any significant overhead with respect to the latency of the plain CCM implementation. The bandwidth was correctly aggregated: it grows from 9.8 MB/s for the C++ implementation (resp. 8.3 MB/s for the JAVA implementation) for a one-node to one-node parallel component configuration to 78.4 MB/s (resp. 66.4 MB/s) for a 8-node to 8-node parallel component configuration. These number have been obtained using a Fast-Ethernet network for CORBA communications.

When using PADICOTM to route CORBA communications through a Myrinet network, the bandwidth for the C++ version scales from 43 MB/s (one-node to one-node) to 280 MB/s (8-node to 8-node). These numbers are better than for a Fast-Ethernet network. However, there are not very good with respect to the performance of the Myrinet network. As we have shown with the PADICOTM experiments, the problem lies in the data copies generated by Mico which limit the bandwidth. High performance parallel components, like high performance parallel objects, require a high performance CORBA implementation. OmniORB is such an implementation for CORBA objects. Such a CORBA component implementation is still missing.

In the future, we will focus on the relationship between parallel components and the CCM containers. It seems parallel versions of existing containers should be define in order to add parallelism support to container operations.

Another direction concerns the adaptability of parallel components to their environment. An example of adaptation is a modification in the number of components that belong to a parallel component. Adaptability appears as a new type of service brought by parallel containers.

Last, we target to develop a fully operational prototype called GridCCM which will extend CCM with parallel components. It will be based on PACO++ and some tools like a compiler for IDL3 transformation and new code generators.

6.2.4. Parallel component deployment for computational grids

Participants: Sebastien Lacour, Christian Pérez, 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 have started to work on an accurate description of the resources. Previous work succeeds in describing properly the compute nodes (CPU speed, memory size, operating system, etc), but generally fails to describe the network topology and its characteristics in a simple, synthetic and complete way. We have proposed a description model for grid networks. This model provides a *synthetic* view of the network topology. In particular, it is able to describe non-hierarchical topologies. It is also a *simple*, namely thanks to the possibility for a network group to inherit properties from its parent network groups. However, this simplicity does not hinder the description of *complex* network topologies (asymmetric links, firewalls, non-IP networks, non-hierarchical topologies). Finally, our description model aims to be *complete* by specifying the necessary information about the software available to access particular network technologies. The proposed model has been successfully integrated into the MDS2 of Globus. It has mainly consisted in defining around 40 LDAP schema entries that describes natures of networks, lists of open or closed ports, average latencies and bandwidths, network software information, etc.

We foresee two complementary future work. On one hand, we have to specify the interactions between a Grid middleware such as OGSA and the CORBA component model. On the other hand, we have to integrate our parallel component model to some planner tools such as Sekitei.

6.2.5. Adaptive components

Participants: Jérémy Buisson, Jean-Louis Pazat.

Key words: *metacomputing*, *Java*, *framework*, *objects*, *components*.

In the area of wireless computing where resources are a key issue, many techniques of dynamic adaptation have been developed: from the observation of the environment, codes can adapt their behavior to fit the resource constraints. An efficient way to allow an application to evolve according to its environment is to provide mechanisms that permit dynamic self-adaptation by changing the behavior depending on the currently available resources. Since Grid architectures are also known to be highly dynamic, using resources efficiently on such architectures is a challenging problem too. 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 combined a dynamic adaptation framework with parallelism and distribution allowing its use for Grid programming. Our prototype is built using the ACEEL adaptation engine built for wireless and mobile environments. Our tool takes into account the parallelism that can reside in applications. We have defined a parallel self-adaptable component as a component composed of several processes working together which is able to change its behavior according to the changes of the environment. The structure of such a component includes an adaptation *policy*, a set of available implementations, called *behaviors*, and a set of *reaction steps*. Reaction steps are the means by which the component adapts itself. It can be for example the replacement of the active behavior, the tuning of some parameters, the redistribution of arrays. The platform we have built mainly provides two kinds of objects: the *decider* and the *coordinators*. The decider is the object that makes the decisions: it decides when (events to watch) and how (reactions to execute) the component should adapt itself according to the adaptation policy. The *coordinators* execute the directives given by the decider: they serve as intermediaries between the code of the component and the platform. Their role is to synchronize the adaptation mechanism with the functional code and to coordinate the execution of the reactions.

We plan to define more formally the properties that the component is required to satisfy to adapt itself. This includes the properties of global states where an adaptation can occur and the constraints on behavior replacement. Studying the relationship between fault tolerance systems that use checkpointing and adaptation in the context of Grid computing is an important perspective too. Finding shared properties between checkpoints and adaptation points would be of great help in establishing properties and constraints on adaptation point placement.

Our long term goal is to build a generic platform to develop parallel adaptable components for the Grid. This platform would include both the toolbox to build parallel self-adaptable components and their runtime environment. Such a platform should ease the building of efficient applications for Grid architectures.

6.3. Large-scale data management for grids

6.3.1. Mome e-Toile

Participant: Yvon Jégou.

Providing the data to the applications is a major problem 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 necessary to move the data from the production sites to the execution sites. Moreover, in high performance simulation domain, the applications are themselves parallel programs and the grid sites are clusters of computation nodes. Each process of the parallel application needs only part of the input data and produces a part of the results. Duplicating the input data from a central server and then gathering the results after the execution can be expensive.

The participation of the PARIS Project-Team to the *e-Toile* project (http://www.urec.cnrs.fr/etoile/) aims at the experimentation of Distributed Shared Memory technology for the implementation of uniform data-naming and data-sharing services for grid computing. The current implementation is based on the MOME DSM. A MOME daemon process is launched in the background on each node of the grid. When the execution of some application starts on MOME-aware computation nodes, each of its parallel processes connects to local MOME daemon. The data-repository interface provides entry-points for the creation and for the localization of segments in the DSM (through a kind of directory), and for the mapping of these segments in the local address

space of the process. The data repository is persistent: the segments retain their data after all application processes have disconnected. The application processes can fail safely (or be killed) without impacting the DSM. The system provides a kind of uniform data space to the grid applications.

The MOME DSM behaves as a COMA (Cache Only Memory) for page management: a copy of a page is present on the nodes recently using the page; a page-fault is served directly by one of the nodes with a valid copy of the page. This strategy is well suited for the case where the computation nodes are clusters. The data never transit through a centralized server.

The current version of MOME considers a flat organization of the DSM nodes. On a grid infrastructure, the performance of the communication system inside a grid node (a cluster) is higher than between grid nodes. The DSM should be aware of this structure. A new hierarchical organization of the MOME DSM has been defined and will be implemented in the near future. This new organization will favor local communications: inter-cluster communications will be avoided as long as page-faults can be served locally. This hierarchical structure will also allow the exploitation of the DSM on a large number of nodes (hundreds of DSM nodes).

MOME currently runs in the user space. Its implementation necessitates no modifications of the kernels. But the applications must use specific library calls in order to exploit the MOME data space. In the future, we plan to interface the MOME daemons with the Linux kernel through a kernel module. The DSM space will become accessible through a classical file system interface without modifications of the applications.

6.3.2. *JuxMem*

Participants: Gabriel Antoniu, Luc Bougé, Mathieu Jan.

With JUXMEM, we propose the concept of *data sharing service* for grid computing, 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)*.

These two classes of systems have been designed and studied in very different contexts. In DSM systems, the nodes are generally under the control of a single administration, and the resources are trusted. In contrast, P2P systems aggregate resources located at the edge of the Internet, with no trust guarantee, and loose control. Moreover these numerous resources are essentially heterogeneous in terms of processors, operating systems and network links, as opposed to DSM systems, where nodes are generally homogeneous. Finally, DSM systems are typically used to support complex numerical simulation applications, where data are accessed in parallel by multiple nodes. In contrast, P2P systems generally serve as a support for storing and sharing immutable files.

Our data sharing service targets physical architectures with features intermediate between DSM and P2P systems. We address scales of the order of thousands of nodes, organized as a federation of clusters, say tens of hundred-node clusters. At a global level, the resources are thus rather heterogeneous, while they can probably be considered as homogeneous within the individual clusters. The control degree and the trust degree are also intermediate, since the clusters may belong to different administrations, which set up agreements on the sharing protocol. Finally, we target numerical applications like heavy simulations, made by coupling individual codes. These simulations process large amounts of data, with significant requirements in terms of data storage and sharing.

The main contribution of such a service is to decouple data management from grid computation, by providing *location transparency* as well as *data persistence* in a *dynamic environment*.

In order 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 [21] (for *Juxtaposed Memory*). A detailed description of this architecture is given in [21]. The 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*). In each `cluster` group, a node manages the

memory made available by the providers of the group (the *cluster manager*). Any node (including providers and cluster managers) can use the service to allocate, read or write to data as a *client*. 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. Consistency of replicated blocks is also handled transparently (according to the sequential consistency model, in the current version). In order to tolerate the volatility of peers, a dynamic monitoring of the number of copies of data block is used and new copies are created when necessary, in order to maintain a given redundancy degree. Cluster manager roles are also replicated, to enhance cluster availability.

As a proof of concept, we have built a software prototype using the JXTA [77] generic peer-to-peer framework, which provides basic building blocks for user-defined peer-to-peer services. As a first evaluation, we have measured the influence of the volatility on the overall behavior of the service. Details are given in [21].

6.4. Advanced models for the Grid

Participants: Jean-Pierre Banâtre, Yann Radenac.

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, for example correctness, coordination/cooperation, mobility, load balancing, migration, efficiency, security, robustness, time, reliability, availability, computing/communication ratio, etc.

The models that we investigate are based on multisets. These models are often presented through metaphors which make understanding easier and may provide new sources of inspiration. One well known metaphor is the chemical one but other metaphors can also be considered such as biology (like cells or DNA), animal societies (like ants or bees colonies), etc.

Our present work relies on the chemical reaction paradigm and more precisely on the Gamma model of programming. Our recent contributions, carried out in close cooperation with Pascal Fradet, now at INRIA Rhône-Alpes (Project-Team *POP ART*), 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. Such a facility will be used to express such properties as code mobility. We have called that model, the γ-calculus. This research has allowed us to define a hierarchy of γ-calculi, from the most basic one (multisets, basic γ-expressions) to a very rich higher-order γ-calculus, much richer than the original Gamma model. A paper [25] was presented at the *Workshop on Membrane Computing 2003*, and a full presentation of this work will be available as an INRIA Research Report.

The generalization of multiplicity of multisets of the γ-calculus opens new ways to express coordination. In particular, we are investigating multisets with infinite cardinality and multisets with a negative cardinality. From our first investigations, these properties may allow the expression of very original coordination schemes.

Apart from completing the above research activities, our perspectives concern coordinations in Grid applications and the definition of a chemical object model for Grids programming.

7. Contracts and Grants with Industry

7.1. VTHD ++

Participants: Yvon Jégou, Christian Pérez.

Program: The VTHD Project (http://www.vthd.org/) aims at deploying a broadband IP test platform to develop the technological bricks that will be required to deploy New Generation Internet and Intranet networks.

Starting time: March 2002.

Ending time: March 2004.

Partners: *France Télécom Recherche & Développement* (FT R&D), INRIA, *École Nationale Supérieure des Télécommunications* (ENST), *École Nationale Supérieure des Télécommunications de Bretagne*, IMAG and EURECOM institute.

Support: RNRT funding, Platform program

Project contribution: The PARIS Project-Team is involved in the VTHD ++ *Metacomputing* Sub-Project. We study code coupling and high-bandwidth data transfer between distant clusters. We have demonstrated a sustained transfer rate of 1.9 Gb/s between a PC cluster located in Rennes and another cluster located in Sophia-Antipolis (1000 km far apart) within a coupled numerical simulation.

7.2. e-Toile

Participant: Yvon Jégou.

Program: The *e-Toile* Project (http://www.urec.cnrs.fr/etoile/) aims at deploying a high-performance Grid platform. We plan to investigate extensions to pre-existing Grid software, to make it suitable for HPC production Grids and to evaluate the costs and benefits of running applications on a Grid platform.

Starting time: December 2001.

Ending time: December 2003 (extension until April 2005).

Partners: CEA, CNRS, CS (Communication & Systems), EDF, ENS Lyon (LIP), Université de Versailles Saint-Quentin (PRISM), INRIA (IRISA, ID-IMAG, RESO), Sun France, IBCP Lyon (since May 2002).

Support: RNTL funding, Platform program

Project contribution: The contribution of the PARIS Project-Team to the e-Toile Project focuses on the development of a *Distributed Shared Memory* (DSM) environment for the implementation of a persistent and distributed data repository for the Grid.

7.3. CASPer

Participants: Guillaume Mornet, Jean-Louis Pazat.

Program: The CASPER Project aims at defining a *Web-based computing portal* to use distributed computing resources.

Starting time: October 2002

Ending time: September 2004

Partners: EADS CCR, ALCATEL Space Industries, IDEAMECH, Université de Paris Sud (LRI)

Support: RNTL funding

Project contribution: The PARIS Project-Team defines the overall architecture and implements an OGSA based system for the core services of CASPER.

7.4. Alcatel

Participants: Yvon Jégou, Christine Morin.

Program: The participation of the PARIS Project-Team to the Software-bus project aims at the evaluation of cluster technology and cluster softwares for Internet routers.

Starting time: March 2001.

Ending time: May 2003.

Partners: ALCATEL, INRIA (IRISA), ENS Lyon (LIP).

Support: ALCATEL funding

Project contribution: The PARIS Project-Team has developed a parallel multi-criteria routing algorithm and implemented it on a cluster of PC using the MOME DSM. Cluster technology allows incremental adaption of the computation power (nodes can be added) and hardware redundancy in case of failure. Our implementation of the routing algorithm exploits the checkpointing capacity of MOME in case of the failure of some node: the routing algorithm is automatically restarted from the last checkpoint on the remaining nodes. New nodes can also be added during restart. The MOME-based software demonstrator was delivered to ALCATEL at the end of the collaboration in May 2003.

7.5. Edf

Participants: Christine Morin, Geoffroy Vallée.

Program: The collaboration with EDF R&D aims at designing and implementing an environment and tools for PC cluster management and use in the area of high performance computing.

Starting time: December 1st, 2000

Ending time: November 30th, 2003

Partners: EDF R&D, INRIA (IRISA, RESO)

Support: EDF R&D funding, PhD CIFRE Grant (Geoffroy Vallée)

Project contribution: The work carried out by the PARIS Project-Team relates to the design and implementation of KERRIGHED Single System Image (SSI) operating system for high-performance computing on clusters. In the framework of the EDF project, KERRIGHED configurable global scheduler has been designed and implemented as well as efficient global process management mechanisms to replicate, migrate and checkpoint processes. A development framework facilitating the implementation of dynamic scheduling policies in KERRIGHED has been developed. Experimentations carried out with applications provided by EDF R&D (HRM1D, Aster, Cyrano3, Cathare) have been conducted.

7.6. Dga

Participants: Renaud Lottiaux, David Margery, Christine Morin.

Program: The COCA contract comprises of two parts. The first one aims at designing, evaluating and optimizing a prototype high performance computing infrastructure well-suited for scientific numerical simulation. The second one relates to the problematic of the reusability of numerical models. The PARIS project-team contributes to the first part of the COCA contract.

Starting time: March 1, 2003

Ending time: July 31, 2005

Partners: DGA, CGEY, ONERA-CERT

Support: DGA Funding

Project contribution: The high-performance computing infrastructure considered in the COCA contract is a federation of medium-size clusters, each cluster running a Single System Image (SSI) operating system. The work carried out by the PARIS Project-Team relates to the design and implementation of KERRIGHED SSI cluster operating system. Four successive releases of KERRIGHED will be delivered as part of the COCA contract with an increasing set of functionalities: (1) Global memory management (V0.70); (2) Global management of memory, processes, data streams and files (V1.0); (3) Checkpointing mechanisms for parallel applications (V.1.10, based on V1.0); and (4) Full-fledged SSI, highly available system (V2.0, based on V.1.10). Moreover, the PARIS Project-Team will study extensions to KERRIGHED operating system to make it a *Grid-aware* operating system for cluster federations.

8. Other Grants and Activities

8.1. Regional grants

GRID 5000: the PARIS Project-Team received a 50,000 Euros grant from the Brittany Regional Council for its participation to the GRID 5000 Platform. The Scientific Council of UNIVERSITY RENNES 1 also supported this initiative by allocating an amount of 8,000 Euros through the local BQR Program.

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 André Ribes (starting on October 1, 2001, for 3 years). This support amounts to a total of 28,000 Euros/year.

8.2. National grants

8.2.1. ACI Globalisation des Ressources Informatiques et des Données

The PARIS Project-Team is deeply involved in national initiatives related to the Grid. An initiative was launched by the *Ministry of Research* through the ACI program (*Action Concertée Incitative*). The ACI GRID (for *Globalisation des Ressources Informatiques et des Données*) aims at fostering French research activities in the area of Grid computing by providing financial support to the best research groups. The ACI GRID initiative was launched in 2001 and issued three calls for proposal (one every year). The PARIS Project-Team submitted proposals for each of them. The following paragraphs present an overview of the projects funded by the ACI GRID in which the project-team is involved.

8.2.2. ACI GRID RMI

Participants: Alexandre Denis, Yvon Jégou, Jean-Louis Pazat, Christian Pérez, Thierry Priol, André Ribes.

The goal of this project is to promote a programming model for computing Grids. This model should combine both parallel programming and distributed programming models. It is based on the concept of distributed objects and software components for distributed programming. This project also aims at designing and experimenting a high-performance communication software framework, enabling both efficient communication between objects or components, and parallel programming. This 2-year project, coordinated by the PARIS Project-Team, ends November 2003. The other partners are the Runtime Project-Team in Bordeaux, the Jacquard Project-Team in Lille, and the Oasis Project-Team in Sophia-Antipolis.

8.2.3. ACI GRID HydroGrid

Participants: Christian Pérez, André Ribes.

The HydroGrid project is a 3-year multidisciplinary project, started in September 2002. It aims at modeling and simulating fluid and solute transport in subsurface geological media using a multiphysic approach. Such multiphysic numerical simulation involve code featuring different languages and communication libraries

(FORTRAN, MPI, OpenMP, etc.), to be run on a commun computational Grid. Therefore, the project relies on the results of the ACI GRID RMI project. A strong point of the HydroGrid project is to group together teams with different areas of expertise (from applications, scientific computing and computer science). The partners are the PARIS, Aladin and Estime Project-Teams at IRISA, the *Hydrodynamique et Transferts en Milieux Poreux* Team (IMFS Strasbourg) and the *Transferts physiques et chimiques* Team (Géosciences Rennes).

8.2.4. ACI GRID DataGRAAL

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

This project gathers together the national research communities interested in large-scale data management. This includes communities involved in grid computing, operating systems, distributed systems and databases. The projects aims at understanding the common research issues and at defining a common terminology. An important goal of the project is encourage the emergence of software prototypes resulting from collaboration efforts of these communities.

The setup of the GDS Project of the ACI MD, coordinated by the PARIS project-team, in collaboration with the ReMaP/GRAAL and REGAL Research Groups, is a result of the discussions that took place within the framework provided by DataGRAAL. The DataGRAAL community is the initiator of a project for a Spring School on *large-scale data management* (DRUIDE 2004), which will take place in May 2004 at Port-aux-Rocs (Le Croisic, Brittany). Gabriel Antoniu chairs the Program Committee and the Organizing Committee of this school. The project started in November 2002 and will end in November 2004. Gabriel Antoniu is local correspondent of DataGRAAL for the PARIS Project-Team.

8.2.5. ACI GRID GRID2

Participants: Jean-Louis Pazat, Christian Pérez.

Jean-Louis Pazat is at the head of the GRID2 project. At many as 10 laboratories from various parts of France are involved in this 150,000-Euro project granted by the Ministry of Research for 3 years. Christian Pérez is in charge of the *Run-Time System and Middleware* Working Group. The objective of this project is to federate the Computing GRID research community by organizing meetings between researchers, teaching for young researchers and by achieving information dissemination.

GRID2 is divided in the following working groups: (1) Software architecture and languages; (2) Run-time systems and middleware; (3) Algorithms and models; (4) Algorithms and high performance applications. This project has organized a *Winter School on Grid Computing* in Aussois in December 2002 and two workshops during the *RenPar* Conference in 2002 and 2003. A number of *Hands-On Days* have taken place this year, enabling researcher to gain practical experience of topics such as *JXTA*, CORBA, numerical computing, etc.

8.2.6. ACI GRID Alta

Participants: Alexandre Denis, Christian Pérez.

Alta is a 2-year joint project funded by the ACI GRID of the French Ministry of Research, in cooperation with the INRIA Cooperative Research Initiatives. The PARIS Project-Team coordinates the project. It also involves *Runtime* Project-Team in Bordeaux, and the *Distribution and Parallelism* Team in Lille. It aims at studying the impact of tolerant loss-control in the context of asynchronous iterative algorithm. An objective is to define and to implement a dedicated API.

8.2.7. ACI GRID Core Grid

Participant: Thierry Priol.

This project (http://www.irisa.fr/CoreGRID/) aims at helping the PARIS Project-Team to prepare a *Network Of Excellence* proposal in the area of *Grid and Peer-to-Peer Computing*. Four European meetings were organized in 2003 and supported by this program: Rennes, April 15-16; Paris, July 15-16; Klagenfurt, August 27; Venice, September 1. These meetings led to the redaction of the CORE GRID proposal that was submitted to the 2nd Call (October 15) of the IST Program (Framework Programme 6, European Commission) under Strategic Objective 2.3.2.8 *Grid-based systems for Complex Problems Solving*.

8.2.8. ACI GRID GRID5000

Participant: Yvon Jégou.

GRID5000 is a nation-wide initiative to build a research platform (ca. 5000 processors) for Grid computing. This large-scale distributed platform will enable experimentations on operating systems, middlewares, and communications libraries by the computer-science research community in France. In 2003, the PARIS Project-Team submitted a proposal for building a GRID5000 node in Rennes. The project has been selected by the French Ministry of Research (ACI GRID) to be one of the 7 initial nodes of the Grid5000 Computing Infrastructure and received a three-year grant of 200 kEuros. The integration of the first 66 processor boards (dual Xeon) to the PARIS cluster (50 PC and Xserve dual-processor nodes) has been initiated during November 2003.

8.2.9. ACI Masses de Données

The PARIS project-team is involved in the ACI MD (for *Masses de Données*). 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.

8.2.10. ACI MD GDS

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

The GDS Project of the 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 will be built using the generic JUXMEM platform for peer-to-peer data management (currently under development within the PARIS project-team, see section 6.3.2). JUXMEM will serve to 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 will end in September 2006. It is coordinated by Gabriel Antoniu (PARIS). Project site: http://www.irisa.fr/GDS/.

8.2.11. ACI MD MDP2P

Participant: Yvon Jégou.

The main objective of the ACI MD MDP2P project is to provide high-level services for managing text and multimedia data in *large-scale P2P systems*. The PARIS Project-Team contributes for the development of DSM-based (MOME and KERRIGHED) data management techniques on clusters of clusters for large-scale multimedia indexing.

8.2.12. ACI MD GdX

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

The *Data Grid Explorer* (GdX) Project aims to implement 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 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 ACI MD, coordinated by PARIS, is partner of GdX, as a user project. The project started in September 2003 and will end in September 2006. Gabriel Antoniu is local correspondent of GdX for the PARIS Project-Team.

8.2.13. Other grants

8.2.14. ARC RedGrid

Participants: Yvon Jégou, Christian Pérez, Thierry Priol, André Ribes.

This 2-year project is funded by the INRIA Cooperative Research Initiative (ARC) whose partners are the ReMaP, PARIS, Algorille and Scalapplix Project-Teams. Its objective is to study the issues related to data

redistribution in a Grid environment, to develop data redistribution libraries and to apply the results in the environments develop by the partners (DIET, PACO++, GridCCM and EPSN).

8.2.15. CNRS AS 114, RTP 8

Participant: Yvon Jégou.

This one-year project, called *Étude préparatoire pour une plate-forme de grille expérimentale*, aims to identify issues (scientific and technical) and propose solutions in the perspective of building an experimental Grid platform gathering nodes geographically distributed in France.

8.2.16. CNRS AS 115 RTP 8

Participant: Christian Pérez.

This is a one-year project, called *Méthodologies de programmation des grilles*, that aism at identifying future research directions related to computational Grids programming. It encompass partners involved in applications, algorithms, runtimes/middleware systems and network protocols.

8.3. European grants

8.3.1. IST POP

Participants: Yvon Jégou, Christian Pérez.

The POP Project (IST Project 2000-29245) targets performance portability of OpenMP application. It is a 3-year project which has started in December 2001. The partners are the *European Center of Parallelism of Barcelona* (CEPBA-UPC, Barcelona, Spain), the *Istituo di Cibernitica* (IC-CNR, Naples, Italy), the *High Performance Information System Laboratory* (LHPCA-UP, Patras, Greece) and INRIA.

The POP Project was motivated by the adoption by the industry of the OpenMP language as a standard for shared memory programming. However, this standard is restricted to hardware shared memory machine. The POP project objective is to build an environment that, starting from an OpenMP application, is able to generate efficient code for different kind of machine architectures. In addition to hardware shared memory machine, the targeted architectures include distributed memory machines and multithreaded machined.

In particular, the project focus on three main goals. The first goal deals with the extension of OpenMP expressiveness to exploit parallelism in irregular task graphs, to improve work-distribution schemes among groups of processors so as to enforce data locality and to add a support for inspector/executor techniques. The second goal is to study the dynamic adaptability of the runtime to use self-analysis to modify the behavior of the application on runtime and to run the same binary file regardless of the underlying architecture, the input data, and the dynamic variation of available resources. The third goal concerns architectural modifications to efficiently execute OpenMP application on distributer memory machine or multithreaded machine.

The POP Project is based on the results of the Nanos European project. In particular, an OpenMP compilation and execution environment was developed for shared memory machines like the Origin 2000.

The PARIS Project-Team focus on the architectural modifications of existing software DSM to provide an adequate support for an efficient execution of OpenMP application on cluster. The set of critical SDSM features, we have identified during Year 2002, are being applied to the MOME SDSM, used in conjunction with PADICOTM. A first complete prototype of the POP Runtime is available on top of MOME.

8.3.2. Core Grid

Thierry Priol is the Scientific Coordinator of a *Network of Excellence* proposal, called CORE GRID, in the area of Grid and Peer-to-Peer (P2P). This proposal was submitted on October 15, 2003. As many as 42 partners, mostly from European countries are involved. The CORE GRID 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. It is the primary objective of the CORE GRID Network of Excellence 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, and more specifically distributed systems and middleware, programming models, knowledge discovery, intelligent tools, and environments.

The research program is structured around five complementary research areas 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 model, system architecture, resource management and scheduling, problem solving environments, tools and Grid systems.

8.3.3. GridCoord

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.

Currently several Grid Research initiatives are on-going or planned at national and European Community level. These initiatives propose 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 is namely to achieve such a coordinated approach. It will require both: (1) Co-ordination among the funding authorities; (2) Collaboration among the individual researchers; (3) A visionary research agenda. This proposal is thus tightly connected to the CORE GRID Network of Excellence proposal above, led by Thierry Priol at the European level.

The GRIDCOORD SSA proposal is led by Marco Vanneschi, University of Pisa, Italy. It includes 10 partners from

The participation of the PARIS Project-Team in the GRIDCOORD proposal from 6 European countries. The French partners are INRIA (Leader: Michel Cosnard) and ENS CACHAN (Leader: Luc Bougé).

8.4. International bilateral grants

8.4.1. Europe

University of Ulm, Germany. A proposal for a bi-lateral research collaboration with the distributed system group of the University of Ulm has been submitted to the 2004 *Procope* Program. During the collaboration, we will study, design and implement new checkpointing strategies for real applications running in different DSM environments. Three DSM systems will be considered: *Plurix* system, developed at the University of Um, which is based on a DSM implemented at the lowest possible level; the KERRIGHED system, which implements a kernel-level DSM in Linux; and the MOME DSM, implemented in user space on top of Linux. The two latter systems are developed in the PARIS Project-Team.

8.4.2. North-America

NSF/INRIA contract, Univ. New Hampshire, USA. This funding has supported our collaboration with the Parallel Computing group of the CS Department of Univ. of New Hampshire (Phil Hatcher and Bob Russell, Professors at UNH). The collaboration has focused on the *Hyperion* project, a distributed compiling and execution environment for clusters. Gabriel Antoniu, Mathieu Jan and Sebastien Monnet have visited the UNH team for 10 days in October 2003, in order to discuss about further collaboration topics related to large-scale data management. It has been decided that David Noblet (one of Phil Hatcher's undergraduate students) will visit IRISA in May 2004 for a 2-month internship within the PARIS Project-Team. He will be supervised by Luc Bougé and Gabriel Antoniu. The internship will be funded by the *International Research Opportunities Program* (IROP) de UNH (http://www.unh.edu/urop/discoverirop.html).

Rutgers University, USA. Pascal Gallard has spent 3 months (May–August 2003) at Rutgers University in the *Discolab* Research Team leaded by Liviu Iftode. This internship has been funded by *Discolab*. Pascal Gallard has worked on the use of the R-DMA technology for the implementation of a highly-available cluster architecture. He contributed to the design of transparent migration mechanisms for TCP/IP streams, that do not require any modification of the TCP/IP protocol or in the client. A prototype has been developed for a PC cluster based on the Myrinet XP networking technology. Liviu Iftode visited PARIS in November 2003 to discuss about a further collaboration on the design and implementation of a novel, highly-available cluster architecture based on the concept of *remote healing*.

8.4.3. Middle-East, Asia, Oceania

Indian Institute of Technology, Kharagpur. Ramamurthy Badrinath, assistant professor at the IIT Kharagpur, has been funded by the MAE and INRIA as an invited researcher in the PARIS Project-Team from May 2002 to May 2003. He has worked with Christine Morin and Geoffroy Vallée on the design of basic fault tolerance building blocks to support various backward error recovery strategies for different kinds of parallel applications executed in a cluster. The proposed mechanisms have been implemented as part of the KERRIGHED single system image operating system and experimented for multithreaded applications.

Seoul National University, Korea. The PARIS Project-Team, with the ReMaP/GRAAL Project-Team located at INRIA Rhône-Alpes, have been selected by the STAR program of the French Embassy in Seoul to conduct a 2-year cooperation with the Department of Aerospace Engineering (Prof. Seung Jo Kim) of the Seoul National University. This cooperation, starting in June 2003, aims at experimenting a Grid infrastructure, made with the computing equipments of the two participants, with aerospace applications (SNU) and middleware and programming tools designed by INRIA. Four researchers from the two INRIA project-teams visited SNU in December 2003 to start the cooperation and attended a French-Korea workshop.

Indian Student Internship. The PARIS Project-Team hosted the 3-month internship (May–July 2003) of Maka Hitashyam from IIT Kharagpur, in the framework of the INRIA *International Internship* Program. Maka Hitashyam worked on the implementation of the KERRIGHED Monitor, a tool to help programmers to tune parallel applications executed on a cluster running KERRIGHED Single System Image operating system.

8.5. Visits and invitations

Ulm University, Germany. Michaël Schöttner, Junior Professor at the University of Ulm, Germany, was invited by the PARIS Project-Team in January 2003 and presented the *Plurix* Project at the IRISA Network and System Seminar.

Rutgers University, USA. Liviu Iftode, Associate-Professor at Rutgers university and Head of the *Discolab* Laboratory, visited us in November 2003 to discuss about further cooperation about self-healing cluster operating systems and to work on a joint proposal on this subject to be submitted to the NSF/INRIA bi-lateral program.

Oak Ridge National Laboratory (ORNL), USA. Stephen Scott is a Senior Researcher at Oak Ridge National Laboratory. He was invited by the PARIS project in October 2003. The goal of his visit was to discuss about a collaboration between INRIA, EDF R&D and ORNL for the integration of KERRIGHED into OSCAR (http://oscar.openclustergroup.org/), a distribution for high performance computing on clusters. KERRIGHED would be integrated as a new package, SSI-OSCAR. Stephen Scott also presented the activities of his research group and OSCAR distribution at the IRISA Network and System Seminar.

Universidade Nova de Lisboa, Portugal. Paulo Afonso Lopes is a PhD student under the supervision of Prof. Dr. José C. Cunha, in the Distributed System Group of the Computer Science Department of the *Universidade Nova de Lisboa*, Portugal. He visited the PARIS Project-Team for 3 days in November 2003. The goal of his visit was to discuss about high performance I/O in clusters with Gaël Utard and persons involved in the design and implementation of KERRIGHED. Paulo Lopes intends to use the KERRIGHED software in the high-performance I/O system prototype he will develop during his thesis.

9. Dissemination

9.1. Community animation

9.1.1. Leaderships, Steering Committees and community service

CNRS, GDR ARP. L. Bougé chairs the CNRS Research Co-operative Federation (*Groupement de recherche*, GDR) on Architecture, Networks and Systems, and Parallelism (ARP, http://www.arp.cnrs.fr/). He has been serving since Year 2000. This GDR GDR is run by the STIC CNRS Department. It is one of the 6 nation-wide animation networks (*GDR d'animation*) run by the Department. It has been renewed for another 4-year term in 2002. Virtually all the French academic researchers active in these areas are registered in the GDR. As of today, this amounts to ca. 900 persons.

J.-L. Pazat is the coordinator of the G2C (*Grids and Clusters for Computing*) Working Group of GDR ARP. This working group aims at information dissemination and contacts between researchers in the area of Cluster and Grid computing.

CNRS, Inter-GDR Co-ordination Committee. L. Bougé chairs the (informal!) Co-ordination Committee of the 6 GDR of the CNRS STIC Department. He has been serving since Year 2001.

ACI GRID, Ministry of Research. L. Bougé et Th. Priol are members of the Scientific Committee of the ACI GRID. The ACI GRID is the national French initiative in the area of Grid computing (http://www.recherche.gouv.fr/recherche/aci/grid.htm). It is led by Michel Cosnard, INRIA Sophia. This initiative was launched in April 2001. Since then, several call for proposals (one per year) were issued, and evaluation was carried out by the Scientific Committee.

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

RenPar Annual Conference. J.-L. Pazat serves as the Chair of the Steering Committee of the RenPar (*Rencontres francophone du parallélisme*, http://www.renpar.org/) annual conference series. The last edition of RenPar was held in La Colle-sur-Loup, near Nice, in 2003.

Linux Cluster Workshop. Ch. Morin co-organized with Jean-Yves Berthou, EDF R&D, a workshop on *Operating systems, tools and methods for high-performance computing on Linux clusters*. It was held in Clamart in October 2003, in the framework of the EDF R&D conference series *Printemps de la recherche*. About 180 persons attended the workshop (http://www.irisa.fr/manifestations/2003/LinuxClusters/).

IPDPS 2003 Conference. L. Bougé is a member of the *Steering Committee* of the IPDPS (*International Parallel and Distributed Processing Symposium*, http://www.ipdps.org/) annual conference series. Together with Michel Cosnard, INRIA Sophia, they co-organized the 2003 Edition in Nice in April 2003, supported by the INRIA Research Unit of Sophia-Antipolis, the I3S CNRS Research Unit and the University of Nice. More than 500 researchers attended.

ICS 2004 Conference. Several members of the PARIS Project-Team are involved in the Organization Committee of the *18th ACM International Conference on Supercomputing* (ICS '04). It will be held in Saint-Malo, France, in June 2004, co-supported by IRISA/INRIA and ENS Lyon. Ch. Morin holds the Local Arrangement Co-Chair, L. Bougé the Finance Chair, Ch. Pérez the Publication Chair, and Th. Priol the Workshop Chair. About 150 participants are expected (http://graal.ens-lyon.fr/ICS04/).

ACI GRID GRID2. J.-L. Pazat heads the GRID2 Project http://www.irisa.fr/grid2/ of ACI GRID. This project is devoted to dissemination and co-ordination of academic French research groups interested in Grid computing.
Ch. Pérez serves co-ordinates of the *Communication and middleware systems* Working Group in the GRID2 project.

ACI GRID RMI. Ch. Pérez heads the ACI GRID RMI Project. It started in December 2001, for 2 years (http://www.irisa.fr/Grid-RMI/).

ACI GRID/INRIA Alta. Ch. Pérez heads the Alta Project, co-supported by ACI GRID and INRIA. Alta started in 2003, for 2 years (http://www.irisa.fr/alta/).

ACI MD GDS. G. Antoniu heads the GDS (*Grid Data Service*) Project supported by ACI MD. GDS started in September 2003, for 3 years (http://www.irisa.fr/GDS/).

ACI MD GdX. G. Antoniu is the local correspondent of the GdX (*Data Grid Explorer*) Project supported by ACI MD. GdX started in September 2003, for 3 years (http://www.lri.fr/~fci/GdX/).

CORE GRID NoE. Th. Priol is the European *Scientific Coordinator* of the CORE GRID Proposal (http://www.irisa.fr/CoreGRID/). This proposal was submitted for the Second Call of the IST Program of the European Commission.

European IST POP. Ch. Pérez is the INRIA Scientific Correspondent of the European IST POP project, started in December 2001 for 3 years.

RTP STIC CNRS. L. Bougé is a member of the Steering Committee of CNRS Thematic Committee (RTP 8) *High-Performance and Distributed Computing* led by Yves Robert, Lyon, and Brigitte Plateau, Grenoble.

AS STIC CNRS. Ch. Pérez is the local correspondent of CNRS Specific Action (AS 115) of RTP 8 *Methodology of Grid programming*, let by Raymond Namyst, Bordeaux.

9.1.2. Editorial boards, steering and program committees

Christine Morin served in the Program Committees for the following conferences:

- *International Workshop on Distributed Shared Memory on Clusters* (DSM 2003), organized in conjunction with IEEE International Symposium on Cluster Computing and the Grid (CCGrid '03), Tokyo, Japan, May 2003.
- *IEEE International Conference on Distributed Computing Systems* (ICDCS-23), Providence, RI, USA, May 2003.
- *6th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing* (ISORC '03), Hakodate, Hokkaido, Japan, May 2003.
- *2nd International Conference on Web-Based Learning* (ICWL 2003), Melbourne, Australia, August 2003.
- *International Workshop on Distributed Shared Memory on Clusters* (DSM 2004), organized in conjunction with IEEE International Symposium on Cluster Computing and the Grid (CCGrid '04), Chicago, IL, USA, May 2004.

- *18th ACM International Conference on Supercomputing* (ICS '04), Saint-Malo, France, June 2004.

J.-L. Pazat served in the Program Committees of the following conferences:

- *ParCo 2003*, Dresden, September 2003.
- *RenPar 15*, La Colle sur Loup, October 2003.

Th. Priol was one of the two co-chairs of Topic *Grid Computing and Middleware Systems* of the Euro-Par 2003 Conference that was held in Klagenfurt, Austria, in August 2003. He will be the global chair of this topic for Euro-Par 2004, to be held in Pisa, Italy, in August 2004.
Th. Priol is member of the Editorial Board of the *Parallel Computing* journal.
He served in the Program Committees of the following conferences:

- *11th Euromicro Conference on Parallel, Distributed and Network-based Processing*, Genova, February 2003.
- *Workshop on Innovative Solutions for Grid Computing* (InnoGrid), Melbourne, Australia, June 2003.
- *International Conference on Parallel Processing* (ICPP), Kaohsiung, Taiwan, October 2003.
- *Intl. Symposium on High-Performance Computing* (ISHPC-V), Tokyo, Japan, October 2003.
- *2nd European Across Grids Conference*, Nicosia, Cyprus, January 2004.
- *IEEE International Symposium on Cluster Computing and the Grid* (CCGRID), Chicago, IL, USA, April 2004.
- *First International Workshop on Programming Paradigms for Grids and Metacomputing Systems*, Krakóow, Poland, June 2004.
- *International Meeting on High Performance Computing for Computational Science* (VecPar), Valencia, Spain, June 2004.
- *Fifth EuroGraphics Workshop on Parallel Graphics and Visualization*, Grenoble, France, June 2004.
- *Third Workshop on Advanced Collaborative Environments*, Seattle, WA, USA, June 2004

L. Bougé belongs to the *Editorial Advisory Board* of the *Scientific Programming* Journal, IOS Press.
He chairs the Program Committee of Topic 18 *Peer-to-Peer Computing* of the Euro-Par 2003 Conference in Klagenfurt, Austria, August 2003.
He will chair the Program Committee of the *International Conference on High Performance Computing* (HiPC 2004), to be held in Bengalore, India, in December 2004.
He served in the Program Committees for the following conferences:

- *8th International Workshop on High-Level Parallel Programming Models and Supportive Environments* (HIPS 2003), held in conjunction with the IPDPS 2003 Conference, Nice, April 2003.
- *15e Rencontres francophones du parallélisme* (RenPar 15), La Colle-sur-Loup, October 2003.

9.1.3. Evaluation committees, consulting

Ch. Morin has been solicited by the Selection Committee (*Commission de spécialistes*, CSE) for Computer Science, UNIVERSITY RENNES 1, as an external reviewer.

J.-L. Pazat is a member of the Selection Committee (*Commission de spécialistes*, CSE) for Computer Science at University Paris Sud (UPS, Orsay) and University of South Brittany (UBS, Vannes).

Th. Priol has been one of the two co-chairs of an *Expert Group* convened by the DG Information Society of the European Commission (EC) to outline a vision for Grid research priorities over the coming 5 to 7 years (ftp://ftp.cordis.lu/pub/ist/docs/ngg_eg_final.pdf).
He was a member of an *International Co-operative Working Group* on Grid research infrastructures convened by the DG Information Society of the EC to for the creation of a framework for international cooperation in the area of Grid computing.
He served as a reviewer to the following EC-funded projects: IST DAMIEN, EUROGRID and P2PEOPLE. He was also involved in the evaluation of IST proposals.

L. Bougé serves in the *Evaluation Committee* of the RNTL Program.
He serves in the *Expert Group* convened by the Ministry of Research (MSTP 9) to review the application for the Doctoral Supervision and Research Awards (*Primes d'encadrement doctoral et de recherche*, PEDR).
He serves in the *Scientific Committees* of ACI GRID and ACI MD Programs of the Ministry of Research.

Ch. Pérez has served as an external expert for the following programs: RNTL and ACI MD programs of Ministry of Research, Regional funding program of the Franche-Comté Region Authority, Post-Doctoral Grant program of INRIA.

9.2. Academic teaching

Ch. Morin is responsible for a graduate teaching module *High Performance Computing on Clusters and Grids* of the Master Program, UNIVERSITY RENNES 1. Within this module, she gave lectures on distributed operating systems for clusters.
She gave a lecture on clusters, taught in the final year of the *Network Architecture* Track at the Institut National des Télécommunications (INT) in Évry in October 2003.
She gave lectures on synchronization, I/O and file systems within the *Operating system* module, taught for the students of the Software Engineering Diploma, UNIVERSITY RENNES 1.

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.

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. Within this module, he gave lectures on Enterprise Java Beans.

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 gave a lecture in the course *Multidisciplinary object-oriented programming* of the *École Doctorale* of the ENS CACHAN.
He has given a seminar entitled PADICOTM*: a component based infrastructure for Grid Computing* at ENS Lyon, *Magistère d'Informatique et Modélisation* (MIM).
He gave tutorials on *Object-Oriented Middleware and Components for the Grid* at two conferences: *Middleware 2003* (Rio de Janeiro, Brazil, June 2003) and *IPDPS 2003* (Nice, France, April 2003). The tutorials were jointly presented with Denis Caromel (Oasis Project-Team, INRIA Sophia).

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

L. Bougé leads the Master Program in Computer Science at the Brittany Extension of ENS CACHAN (*Magistère Informatique et Télécommunications*, 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.

9.3. Conferences, seminars, and invitations

Only the events not listed elsewhere are listed below.

CORBA components, and JXTA Tutorial Day. G. Antoniu, Ch. Pérez and J.-L. Pazat have organized two tutorial days to introduce the CORBA components and the JXTA peer-to-peer environment (http://www.irisa.fr/grid2/). The tutorials took place in March 2003. They have been supported by GDR ARP and by the GRID2 project of the ACI GRID. About 30 participants attended.

CEA Seminar. G. Antoniu and L. Bougé were invited and gave a talk at the CEA seminar organized in Saint-Malo on *Large-scale data management* in June 2003. Title: *Large-Scale Data Management: a Peer-to-Peer Approach Based on the JXTA Environment*.

Dagstuhl Seminar. G. Antoniu and L. Bougé were invited and gave talks at the Dagstuhl Seminar on *Hardware and Software Consistency Models: Programmability and Performance* in October 2003. Talks: *Peer-to-Peer Distributed Shared Memory?* (G. Antoniu), *Hierarchy-Aware Consistency Protocols for Large-Scale DSM* (L. Bougé).

LCS Seminar, MIT, Boston. G. Antoniu gave an invited talk for Prof. Arvind's group at LCS (MIT, Boston) in October 2003. Title: *Peer-to-Peer Distributed Shared Memory?*.

University of New Hampshire ACM Seminar, Durham. G. Antoniu gave an invited talk at a seminar of the local ACM organization of the University of New Hampshire in October 2003. Title: *Peer-to-Peer: Getting Serious?*.

Sun Microsystems, Santa Clara. G. Antoniu visited the JXTA Team of Sun Microsystems in Santa Clara, CA, where he presented the current status of the JUXMEM Project, in November 2003. Title: JUXMEM*: a JXTA-Based Service for Data Sharing on the Grid*.

EuroPVMMPI'03. Th. Priol gave an invited talk entitled *Programming High Performance Applications using Components* at the 10th European PVM/MPI Users Group conference in Venice, Italy in September 2003.

CERFACS Seminar. Ch. Pérez gave an invited talk entitled *Programming the Grid with parallel software components* at the Sparse Days and Grid Computing at Saint-Girons, organized by the CERFACS and the ENSEEIHT-IRIT.

Apple. Ch. Pérez gave an invited talk entitled *Parallel computing and Clusters* at the Conference on cluster computing organized by Apple at the Apple Executive Briefing Center, Paris, France, December 2003.

University of Cork, Ireland. Ch. Morin gave an invited talk in the Computer Science Department, at UCC, Cork, Ireland, in February 2003. Title: KERRIGHED*: a Single System Image Operating System for High Performance Computing on Clusters*.

SOS7 workshop, Sandia Labs. Ch. Morin was invited to participate to the *7th Workshop on Distributed Computing* (SOS7) organized by Sandia Labs in Durango, Colorado, and to present a talk entitled *Design and Implementation of a Single System Image Operating System for High Performance Computing on Clusters* in March 2003.

Linux Cluster Workshop, EDF. Ch. Morin gave a talk at the Linux Clusters Workshop organized at EDF R&D in Clamart on *Operating Systems, Tools and Methods for High Performance Computing on Linux Clusters* in October 2003. Title: *Single System Image OS for Clusters:* KERRIGHED *Approach*.

University of Ulm, Germany. Ch. Morin gave an invited talk at the Computer Science department of the University of Ulm, Germany, in November 2003. Title: KERRIGHED*: a Linux based Operating system to Ease Cluster Programming*.

ID-IMAG, Grenoble. Geoffroy Vallée has been invited to give a talk entitled *Global process Management in* KERRIGHED *Cluster Operating System* at ID-IMAG, Grenoble, in June 2003.

LIP, ENS Lyon. Geoffroy Vallée has been invited by the RESO INRIA Project-Team to give a talk entitled *Global process Management in* KERRIGHED *Cluster Operating System* at LIP, ENS Lyon, June 2003.

9.4. Administrative responsibilities

J.-P. Banâtre is in charge of the European Affairs within the Department for European and International Relations (DREI) at INRIA.

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.

Ch. Morin is a member of the INRIA Evaluation Committee.
She chairs the local IRISA Computing Infrastructure User Committee (*Commission des utilisateurs des moyens informatiques,CUMI*).

J.-L. Pazat is a member of the Administrative Committee of INSA of Rennes.
He served as a member of the Administrative Committee of The Computing Resource Center of INSA of Rennes.

Th. Priol was Deputy Director of the INRIA Evaluation Committee till June 2003.

9.5. Miscellaneous

L. Bougé was a member of the 2003 Selection Committee for the Junior Researcher permament position (CR2) at IRISA.
He is a member of the Project-Team Committee of IRISA, standing for the PARIS Project-Team.
He is a deputy-member of the Selection Committee (*Commission de spécialistes*, CSE) for Computer Science at ENS CACHAN.

Ch. Morin is a member of the IRISA *Local Committee Temporay Positions* (*Commission locale des postes d'accueil*, CPLA).
She has been a member of the editorial board of *Inedit*, the INRIA Newsletter since September 2003.
She has been enrolled in September 2003 as an external member of the *Course Advisory Board* of the Information Technology School of Deakin University (Australia).

Th. Priol is a deputy-member of the Selection Committee (*Commission de spécialistes*, CSE) for Computer Science at UNIVERSITY RENNES 1, since December 2001.

10. Bibliography

Major publications by the team in recent years

[1] 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) », series Lecture Notes in Computer Science, volume 919, Springer Verlag, pages 176–183, Milan, Italy, May, 1995.

[2] 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) », series Lect. Notes in Comp. Science, volume 2026, Held in conjunction with IPDPS 2001. IEEE TCPP, Springer-Verlag, pages 55–70, San Francisco, April, 2001, http://www.inria.fr/rrrt/rr-4108.html, Available as INRIA Research Report RR-4108.

[3] J.-P. BANÂTRE, D. L. MÉTAYER. *Programming by Multiset Transformation.* in « Communications of the ACM », number 1, volume 36, January, 1993, pages 98–111.

[4] 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) », IEEE Computer Society, pages 144–151, Berlin, Germany, May, 2002, http://www.inria.fr/rrrt/rr-4554.html, Available as INRIA Reserach Report RR-4554.

[5] A.-M. KERMARREC, C. MORIN, M. BANÂTRE. *Design, Implementation and Evaluation of ICARE.* in « Software Practice and Experience », number 9, 1998, pages 981–1010.

[6] 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 », number 10, volume 44, October, 2001, pages 110–117, http://www.irisa.fr/paris/biblio/Papers/Bouge/KieHatBouBal01CACM.ps.gz, Special issue on Java for High Performance Computing.

[7] 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) », series Lecture Notes in Computer Science, volume 634, Springer Verlag, pages 441–452, September, 1992, http://www.inria.fr/rrrt/rr-1634.html.

[8] T. PRIOL. *Efficient support of MPI-based parallel codes within a CORBA-based software infrastructure.* in « Response to the Aggregated Computing RFI from the OMG, Document orbos/99-07-10 », July, 1999.

Doctoral dissertations and "Habilitation" theses

[9] A. DENIS. *Contribution à la conception d'une plate-forme haute performance d'intégration d'exécutifs communicants pour la programmation des grilles de calcul.* Thèse de doctorat, Université de Rennes 1, IRISA, Rennes, France, December, 2003, http://www.irisa.fr/bibli/publi/theses/theses03.html, À paraître.

Articles in referred journals and book chapters

[10] A. DENIS, C. PÉREZ, T. PRIOL. *Achieving Portable and Efficient Parallel CORBA Objects.* in « Concurrency

and Computation: Practice and Experience », number 10, volume 15, August, 2003, pages 891–909, http://www.irisa.fr/paris/Biblio/Papers/Denis/DenPerPri03CCPE.ps.

[11] A. DENIS, C. PÉREZ, T. PRIOL. *PadicoTM: An Open Integration Framework for Communication Middleware and Runtimes.* in « Future Generation Computer Systems », number 4, volume 19, May, 2003, pages 575–585, http://www.irisa.fr/paris/Biblio/Papers/Denis/DenPerPri03FGCS.pdf.

[12] A. DENIS, C. PÉREZ, T. PRIOL, A. RIBES. *Parallel CORBA Objects for Programming Computational Grids.* in « Distributed Systems Online », number 2, volume 4, February, 2003, http://dsonline.computer.org/0302/f/pri_print.htm, Electronic journal.

[13] P. GALLARD, C. MORIN. *Dynamic Streams For Efficient Communications between Migrating Processes in a Cluster.* in « Parallel Processing Letters », number 4, volume 13, December, 2003, http://www.inria.fr/rrrt/rr-4987.html, to appear.

[14] A.-M. KERMARREC, C. MORIN. *HA-PSLS: a Highly Available Parallel Single Level Store System.* in « Concurrency and Computation: Practice and Experience », number 10, volume 15, July, 2003, pages 911–937, http://www.irisa.fr/paris/Biblio/Papers/Kermarrec/KerMor02ccpe.pdf.

[15] C. MORIN, P. GALLARD, R. LOTTIAUX, G. VALLÉE. *Towards an Efficient Single System Image Cluster Operating System.* in « Future Generation Computer Systems », number 2, volume 20, January, 2004, http://www.irisa.fr/paris/Biblio/Papers/Morin/MorGalLotVal03FGCS.pdf.

[16] C. PÉREZ, T. PRIOL, A. RIBES. *A Parallel CORBA Component Model for Numerical Code Coupling.* in « The International Journal of High Performance Computing Applications (IJHPCA) », number 4, volume 17, 2003, pages 417–429, http://www.irisa.fr/paris/Biblio/Papers/Perez/PerPriRib03IJHPCA.pdf, Special issue Best Applications Papers from the 3rd Intl. Workshop on Grid Computing.

[17] G. VALLÉE, R. LOTTIAUX, L. RILLING, J.-Y. BERTHOU, I. DUTKA-MALHEN, C. MORIN. *A Case for Single System Image Cluster Operating Systems: Kerrighed Approach.* in « Parallel Processing Letters », number 2, volume 13, June, 2003, http://www.irisa.fr/paris/Biblio/Papers/Vallee/ValLotRilBerDutMor03PPL.pdf.

Publications in Conferences and Workshops

[18] F. ANDRÉ, J. BUISSON, J.-L. PAZAT. *Dynamic adaptation of parallel and distributed components on Grid environments.* in « 11th Int. Conference on Advance Computing and Communication », PSG College of Technology, Coimbatore, India, December, 2003.

[19] G. ANTONIU, L. BOUGÉ, M. JAN. *La plate-forme JuxMem : support pour un service de partage de données sur la grille.* in « Actes des Rencontres francophones du parallélisme (RenPar 15) », pages 145–152, La Colle sur Loup, October, 2003, http://www.irisa.fr/paris/Biblio/Papers/Antoniu/AntBouJan03RenPar.ps.gz.

[20] G. ANTONIU, L. BOUGÉ, M. JAN. *Peer-to-Peer Distributed Shared Memory?.* in « Proc. IEEE/ACM 12th Intl. Conf. on Parallel Architectures and Compilation Techniques (PACT 2003), Work in Progress Session », pages 1–6, New Orleans, Louisiana, September, 2003, http://www.irisa.fr/paris/Biblio/Papers/Antoniu/AntBouJan03PACT-WIP.ps.gz.

[21] G. ANTONIU, L. BOUGÉ, M. JAN. *JuxMem: An Adpative Supportive Platform for Data Sharing on the Grid.* in « Proc. Workshop on Adaptive Grid Middleware (AGRIDM 2003) », Held in conjunction with PACT 2003, pages 49–59, New Orleans, Louisiana, September, 2003, http://www.irisa.fr/paris/Biblio/Papers/Antoniu/AntBouJan03PACT-AGRIDM.ps.gz.

[22] G. ANTONIU, L. BOUGÉ, S. LACOUR. *Making a DSM Consistency Protocol Hierarchy-Aware: An Efficient Synchronization Scheme.* in « Proc. Workshop on Distributed Shared Memory on Clusters (DSM 2003) », Held in conjunction with CCGRID 2003, IEEE TFCC, pages 516–523, Tokyo, May, 2003, http://csdl.computer.org/comp/proceedings/ccgrid/2003/1919/00/19190516abs.htm.

[23] R. BADRINATH, C. MORIN. *Common Mechanisms for supporting fault tolerance in DSM and message passing systems.* in « Concurrent Information Processing and Computing », NATO Advanced Research Workshop, Alexandru loan Cuza, University Press, D. GRIGORAS, A. NICALAU, F. L. TIPLEA, editors, pages 37–45, July, 2003, http://www.inria.fr/rrrt/rr-4613.html.

[24] R. BADRINATH, C. MORIN, G. VALLÉE. *Checkpointing and Recovery of Shared Memory Parallel Applications in a Cluster.* in « Proc. Intl. Workshop on Distributed Shared Memory on Clusters (DSM 2003) », Held in conjunction with CCGrid 2003. IEEE TFCC, pages 471–477, Tokyo, May, 2003, http://www.inria.fr/rrrt/rr-4806.html.

[25] J.-P. BANÂTRE, P. FRADET, Y. RADENAC. *Higher-Order Chemistry.* in « Pre-proceedings of Workshop on Membrane Computing (WMC 2003) », pages 53–60, July, 2003, http://www.irisa.fr/paris/Biblio/Papers/Banatre/BanFraRad03WMC.p

[26] A. DENIS. *Architecture d'une plate-forme de communication multi-paradigme pour les grilles.* in « 15es Rencontres francophones du parallélisme (RenPar 15) », INRIA, M. AUGUIN, F. BAUDE, D. LAVENIER, M. RIVEILL, editors, pages 111–118, La Colle sur Loup, France, October, 2003, http://www.irisa.fr/paris/Biblio/Papers/Denis/Den03RenPar.ps.

[27] A. DENIS, C. PÉREZ, T. PRIOL, A. RIBES. *Padico: A Component-Based Software Infrastructure for Grid Computing.* in « 17th International Parallel and Distributed Processing Symposium (IPDPS 2003) », IEEE Computer Society, Nice, France, April, 2003, http://www.irisa.fr/paris/Biblio/Papers/Denis/DenPerPriRib03IPDPS.ps.

[28] A. DENIS, C. PÉREZ, T. PRIOL, A. RIBES. *Bringing High Performance to the CORBA Component Model.* in « SIAM Conference on Parallel Processing for Scientific Computing », February, 2004, http://www.irisa.fr/paris/Biblio/Papers/Perez/DenPerPriRib04PP.txt, To appear.

[29] P. GALLARD, C. MORIN. *Dynamic Streams For Efficient Communications between Migrating Processes in a Cluster.* in « Euro-Par 2003: Parallel Processing », series Lect. Notes in Comp. Science, volume 2790, Springer-Verlag, pages 930–937, Klagenfurt, Austria, August, 2003, http://www.irisa.fr/paris/Biblio/Papers/Gallard/GalMor03europar.ps.

[30] Y. JÉGOU. *Implementation of Page Management in Mome, a User-Level DSM.* in « Proc. Intl. Workshop on Distributed Shared Memory on Clusters (DSM 2003) », Held in conjunction with CCGrid 2003. IEEE TFCC, pages 479–486, Tokyo, Japan, May, 2003.

[31] D. MARGERY, G. VALLÉE, R. LOTTIAUX, C. MORIN, J.-Y. BERTHOU. *Kerrighed: a SSI Cluster OS*

Running OpenMP. in « Proc. 5th European Workshop on OpenMP (EWOMP '03) », September, 2003, http://www.irisa.fr/paris/Biblio/Papers/Margery/Mar03EWOMP.pdf.

[32] C. MORIN, R. LOTTIAUX, G. VALLÉE, P. GALLARD, G. UTARD, R. BADRINATH, L. RILLING. *Kerrighed: a Single System Image Cluster Operating System for High Performance Computing.* in « Proc. of Europar 2003: Parallel Processing », series Lect. Notes in Comp. Science, volume 2790, Springer Verlag, pages 1291–1294, August, 2003, http://www.irisa.fr/paris/Biblio/Papers/Morin/MorLotValGalUtaBAdRil03europar.pdf.

[33] C. MORIN. *Design and Implementation of a Single System Image Operating System for High Performance Computing on Clusters.* in « Proc. 7th Workshop on Distributed Computing », Durango, Colorado, March, 2003, http://www.irisa.fr/paris/Biblio/Papers/Morin/Mor03SOS7.pdf, Invited paper. Electronic proceedings only.

[34] C. PÉREZ, T. PRIOL, A. RIBES. *PaCO++: A parallel object model for high performance distributed systems.* in « Distributed Object and Component-based Software Systems Minitrack in the Software Technology Track of the 37th Hawaii International Conference on System Sciences (HICSS-37) », IEEE Computer Society Press, Big Island, Hawaii, USA, January, 2004, http://www.irisa.fr/paris/Biblio/Papers/Ribes/PerPriRib04HICSS.pdf, To appear.

[35] A. RIBES. *Un modèle orienté objet pour les applications scientifiques distribuées haute performance.* in « 15es Rencontres francophones du parallélisme (RenPar '15) », INRIA, M. AUGUIN, F. BAUDE, D. LAVENIER, M. RIVEILL, editors, La Colle sur Loup, France, October, 2003, http://www.irisa.fr/paris/Biblio/Papers/Ribes/Rib03RP.ps.

[36] G. UTARD, C. MORIN. *Une nouvelle façon d'envisager les entrées/sorties dans un système d'exploitation pour grappe.* in « Actes des Rencontres francophones du parallélisme (RenPar 15) », La Colle sur Loup, October, 2003, http://www.irisa.fr/paris/Biblio/Papers/Utard/UtaMor03RenPar.pdf.

[37] G. VALLÉE, C. MORIN, J.-Y. BERTHOU, L. RILLING. *A New Approach to Configurable Dynamic Scheduling in Clusters based on Single System Image Technologies.* in « Proc. 17th International Parallel and Distributed Processing Symposium (IPDPS 2003) », IEEE, pages 91, Nice, April, 2003, http://www.irisa.fr/paris/Biblio/Papers/Vallee/ValMorBerRil03IPDPS.pdf.

[38] G. VALLÉE. *Un ordonnanceur de processus pour grappe adaptable : mise en oeuvre dans le système Kerrighed.* in « Actes des Rencontres francophones du parallélisme (RenPar 15) », La Colle sur Loup, France, October, 2003, http://www.irisa.fr/paris/Biblio/Papers/Vallee/Val03renpar.pdf.

Internal Reports

[39] F. ANDRÉ, J. BUISSON, J.-L. PAZAT. *Parallel self-adaptable components for the Grid.* Technical report, number 1553, IRISA, 2003.

[40] G. ANTONIU, L. BOUGÉ, M. JAN. *Peer-to-peer distributed shared memory?.* Research Report, number RR-4924, INRIA, IRISA, Rennes, France, September, 2003, http://www.inria.fr/rrrt/rr-4924.html.

[41] G. ANTONIU, L. BOUGÉ, M. JAN. *JuxMem: An Adpative Supportive Platform for Data Sharing*

on the Grid. Research Report, number RR-4917, INRIA, IRISA, Rennes, France, September, 2003, http://www.inria.fr/rrrt/rr-4917.html.

[42] G. ANTONIU, L. BOUGÉ, S. LACOUR. *Making a DSM Consistency Protocol Hierarchy-Aware: An Efficient Synchronization Scheme.* Research Report, number RR-4767, INRIA, IRISA, Rennes, France, March, 2003, http://www.inria.fr/rrrt/rr-4767.html.

[43] R. BADRINATH, C. MORIN. *Locks and Barriers in Checkpointing and Recovery.* Research Report, number RR-5021, INRIA, IRISA, Rennes, France, October, 2003, http://www.inria.fr/rrrt/rr-5021.html.

[44] R. BADRINATH, C. MORIN, G. VALLÉE. *Checkpointing and Recovery of Shared Memory Parallel Applications in a Cluster.* Research Report, number RR-4806, INRIA, IRISA, Rennes, France, April, 2003, http://www.inria.fr/rrrt/rr-4806.html.

[45] A. DENIS, C. PÉREZ, T. PRIOL. *Network Communications in Grid Computing: At a Crossroads Between Parallel and Distributed Worlds.* Research Report, number RR-4975, INRIA, IRISA, Rennes, France, October, 2003, http://www.inria.fr/rrrt/rr-4975.html.

[46] A. DENIS, C. PÉREZ, T. PRIOL, A. RIBES. *Padico: A Component-Based Software Infrastructure for Grid Computing.* Research Report, number RR-4974, INRIA, IRISA, Rennes, France, October, 2003, http://www.inria.fr/rrrt/rr-4974.html.

[47] P. GALLARD, C. MORIN. *Dynamic Streams for Efficient Communications Between Migrating Processes in a Cluster.* Research Report, number RR-4987, INRIA, IRISA, Rennes, France, November, 2003, http://www.inria.fr/rrrt/rr-4987.html.

[48] Y. JÉGOU. *Implementation of Page Management in Mome, a User-Level DSM.* Research Report, number 4999, INRIA, INRIA, France, November, 2003, http://www.inria.fr/rrrt/rr-4999.html.

[49] Y. JÉGOU, C. MORIN. *Recherche de chemins multi-critères, algorithmes séquentiels et parallèles, exécution sur la mémoire partagée répartie Mome, exploitation des points de reprise de Mome en cas de défaillance.* Livrable, number final, Collaboration Alcatel/INRIA, April, 2003, Convention Software-Bus numéro 101C01010031329012.

[50] R. LOTTIAUX, D. MARGERY, G. VALLÉE. *Kerrighed: Reference Manual V0.70.0.* Technical report, number PI-1576, IRISA, November, 2003, http://www.irisa.fr/bibli/publi/pi/2003/1576/1576.html.

[51] R. LOTTIAUX, D. MARGERY, G. VALLÉE. *Kerrighed: Reference Manual V0.72.0.* Technical report, number PI-1577, IRISA, November, 2003, http://www.irisa.fr/bibli/publi/pi/2003/1577/1577.html.

[52] D. MARGERY, G. VALLÉE, R. LOTTIAUX, C. MORIN, J.-Y. BERTHOU. *Kerrighed: a SSI Cluster OS Running OpenMP.* Research Report, number RR-4947, INRIA, IRISA, Rennes, France, August, 2003, http://www.inria.fr/rrrt/rr-4947.html.

[53] S. MONNET, C. MORIN, R. BADRINATH. *Hybrid Checkpointing for Parallel Applications in Cluster Federations.* Rapport de recherche, number RR-5007, IRISA, IRISA, Rennes, France, November, 2003, http://www.inria.fr/rrrt/rr-5007.html.

[54] C. MORIN, P. GALLARD, R. LOTTIAUX, G. VALLÉE. *Towards an Efficient Single System Image Cluster Operating System.* Research report, number RR-5018, INRIA, IRISA, Rennes, France, December, 2003, http://www.inria.fr/rrrt/rr-5018.html.

[55] C. PÉREZ, T. PRIOL, A. RIBES. *PACO++: A parallel object model for high performance distributed systems.* Research Report, number RR-4960, INRIA, IRISA, Rennes, France, October, 2003, http://www.inria.fr/rrrt/rr-4960.html.

[56] L. RILLING, C. MORIN. *A Coherence Protocol for Cached Copies of Volatile Objects in Peer-to-Peer Systems.* Research Report, number RR-5059, INRIA, December, 2003, http://www.inria.fr/rrrt/rr-5059.html.

[57] F. SULTAN, A. BOHRA, Y. PAN, S. SMALDONE, I. NEAMTIU, P. GALLARD, L. IFTODE. *Nonintrusive Failure Detection and Recovery for Internet Services using Backdoors.* Technical report, number DCS-TR-524, Rutgers University, December, 2003, http://discolab.rutgers.edu/bda/index.html, submitted for publication.

[58] G. UTARD, C. MORIN. *Une nouvelle façon d'envisager les entrées/sorties dans un système d'exploitation pour grappe.* Rapport de recherche, number 4948, INRIA, October, 2003, http://www.inria.fr/rrrt/rr-4948.html.

[59] G. VALLÉE, C. MORIN, J.-Y. BERTHOU, L. RILLING. *A New Approach to Configurable Dynamic Scheduling in Clusters based on Single System Image Technologies.* Research Report, number 4801, INRIA, IRISA, Rennes, France, April, 2003, http://www.inria.fr/rrrt/rr-4801.html.

[60] G. VALLÉE. *Un ordonnanceur de processus pour grappe adaptable : mise en oeuvre dans le système Kerrighed.* Research Report, number 4971, INRIA, IRISA, Rennes, France, November, 2003, http://www.inria.fr/rrrt/rr-4971.html.

Miscellaneous

[61] G. ANTONIU, L. BOUGÉ. *Gestion de données à grande échelle : une approche pair-à-pair à partir de l'environnement JXTA.* Comité des utilisateurs informatiques du CEA, Saint-Malo, June, 2003, http://www.irisa.fr/paris/Biblio/Papers/Antoniu/AntBou03CUIC.ppt, Exposé invité.

[62] R. BADRINATH. *A Summary of Work at IRISA May 2002–May 2003.* May, 2003, http://www.irisa.fr/paris/Biblio/Papers/Badrinath/Bad

[63] J. BUISSON. *Adaptation dynamique de composants parallèles pour grilles de calcul.* Rapport de stage de DEA, INSA de Rennes, June, 2003, http://www.irisa.fr/paris/Biblio/Papers/Buisson/Bui03DEA.pdf.

[64] M. JAN. *Architecture d'un service de partage de données modifiables sur une infrastructure pair-à-pair.* Rapport de stage de DEA, DEA d'informatique de l'IFSIC, Université de Rennes 1, France, June, 2003, http://www.irisa.fr/paris/biblio/Papers/Jan/Jan03DEA.pdf.

[65] R. LOTTIAUX. *Bilan du post-doctorat industriel EDF/INRIA.* February, 2003.

[66] S. MONNET. *Conception et évaluation d'un protocole de reprise d'applications parallèles dans une fédération de grappes de calculateurs.* Rapport de stage de DEA, IFSIC, Université de Rennes 1, June, 2003, http://www.irisa.fr/paris/Biblio/Papers/Monnet/Mon03Master.pdf, In French.

[67] Y. RADENAC. *Modèle de calculs et de programmes fondés sur la réécriture de multi-ensembles.* Rapport de stage de DEA, DEA d'informatique de l'IFSIC, Université de Rennes 1, France, June, 2003, http://www.irisa.fr/paris/Biblio/Papers/Radenac/Rad03DEA.pdf.

[68] F. SULTAN, A. BOHRA, P. GALLARD, I. NEAMTIU, S. SMALDONE, Y. PAN, L. IFTODE. *Backdoors: A System Architecture for Nonintrusive Remote Healing.* poster presented at the 19th ACM Symposium on Operating Systems Principles (SOSP), October, 2003, http://discolab.rutgers.edu/bda/bd_poster.gif.

Bibliography in notes

[69] 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.

[70] O. AUMAGE, L. BOUGÉ, A. DENIS, L. EYRAUD, J.-F. MÉHAUT, G. MERCIER, R. NAMYST, L. PRYLLI. *A Portable and Efficient Communication Library for High-Performance Cluster Computing (extended version).* in « Cluster Computing », number 1, volume 5, January, 2002, pages 43–54.

[71] R. BUYYA. *High Performance Cluster Computing: Architectures and Systems.* Prentice-Hall PTR, 1999.

[72] F. DABEK, B. ZHAO, P. DRUSCHEL, I. STOICA. *Towards a Common API for Structured Peer-to-Peer Overlays.* in « 2nd Intl. Workshop on Peer-to-Peer Systems (IPTPS '03) », series Lect. Notes Computer Science, volume 2735, Springer-Verlag, Berkeley, CA, February, 2003.

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

[74] 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.

[75] 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, pages 15–26, May, 1990.

[76] J. GRAY, D. SIEWIOREK. *High Availability Computer Systems.* in « IEEE Computer », September, 1991.

[77] *Project JXTA: Java programmer's guide.* Sun Microsystems, Inc., 2001, http://www.jxta.org.

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

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

[80] L. LAMPORT. *How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs.* in « IEEE Transactions on Computers », number 9, volume 28, September, 1979, pages 690–691.

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

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

[83] D. S. MILOJICIC, V. KALOGERAKI, R. LUKOSE, K. NAGARAJA, J. PRUYNE, B. RICHARD, S. ROLLINS, Z. XU. *Peer-to-Peer Computing.* Research Report, number HPL-2002-57, HP Labs, March, 2002, Submitted to Comuting Surveys.

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

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

[86] A. ORAM. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies.* O'Reilly, 2001.

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

[88] 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) », pages 329–350, November, 2001.

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

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