INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Team KerData

Cloud and Grid Storage for Very Large Distributed Data

Rennes - Bretagne-Atlantique

Theme : Distributed and High Performance Computing

Table of contents

1. Team

Research Scientist

Gabriel Antoniu [Research Associate (CR) INRIA, HDR defended on March 5, 2009, HdR]

Faculty Member

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

PhD Student

Loïc Cudennec [INRIA and Brittany Regional Council Grant, PhD defended on Jan. 15, 2009]
Bogdan Nicolae [MENRT Grant]
Alexandra Carpen-Amarie [INRIA CORDI-S Grant]
Diana Moise [INRIA and Brittany Regional Council Grant]
Viet-Trung Tran [MENRT Grant]

Visiting Scientist

Jesús Montes [PhD student, Polytechnic University of Madrid, 3 months, supported by a Spanish grant]
Alexandru Costan [PhD student, Polytechnic University of Bucharest, 1 month, supported by our bilateral contract]
Jing (Tylor) Cai [Master student, City University of Hong Kong, 5 months, supported by the INRIA Internship Program]
Mihaela Vlad [Master student, Polytechnic University of Bucharest, 4 months, supported by the INRIA Internship Program]

Administrative Assistant

Maryse Fouché [Secretary (TR) INRIA]

Other

Matthieu Dorier [Magistère undergraduate internship, ENS CACHAN Brittany campus, 2 months]
Benjamin Girault [Magistère undergraduate internship, ENS CACHAN, 2 months]

2. Overall Objectives

2.1. Overall Objectives

Our research activities address the area of distributed data management at challenging scales, on grids, clouds, petascale architectures, desktop grids, etc. We target data-oriented high-performance applications that exhibit the need to handle massive non structured data - BLOBs: binary large objects (in the order of terabytes) - stored in a large number of nodes (thousands to tens of thousands), accessed under heavy concurrency by a large number of clients (thousands to tens of thousands at a time) with a relatively fine access grain (in the order of megabytes). Examples of such applications are:

- Grid and cloud data-mining applications handling massive data distributed at a large scale.
- Advanced data storage and management on cloud infrastructures.
- Distributed storage for Petaflop computing applications.
- Data storage for desktop grid applications with high write throughput requirements.
- Distributed data sharing and storage for extremely large databases.

Our current research follows three main research directions.

2.1.1. Multiversion BLOB management

We are currently designing, implementing and experimentally validating a generic data management platform for large-scale distributed infrastructures, called BlobSeer (http://blobseer.gforge.inria.fr/). It is aimed at addressing the challenges mentioned above: huge data, highly concurrent fine-grain access, while supporting versioning and decentralized metadata management.

2.1.2. Scalable BLOB-based distributed file systems

We are exploring how the file system approach can support scalable data management to address the needs of two classes of applications:

- data-mining through massive data using the Map-Reduce paradigm;
- numerical applications for Petaflop architectures.

The goal is to evaluate the benefits of building global file systems using object-based distributed storage as proposed by BlobSeer, which targets efficient, decentralized management of huge data under heavy concurrency.

2.1.3. Monitoring, fault-tolerance and self-steering

We aim at proposing a self-adaptive BLOB management system. To this aim, we are equipping BlobSeer with a number of software sensors to couple it with the MonALISA generic monitoring system (http://monalisa.caltech.edu/). This latter system offers a distributed, modular architecture which can adapt the very large scale of BlobSeer and the high rate of client interactions. MonALISA allows the user to visualize a large number of behavioral parameters in a convenient way. Moreover, it is possible to build a feedback loop from MonALISA to BlobSeer so that BlobSeer can dynamically reconfigure (e.g., in reaction to failures or to a dynamic variation of resource availability), according to the observation of its global behavior by MonALISA. This is the path toward a *self-steering* BlobSeer.

2.2. Highlights

Creation of the KerData Team. The KerData Team was created on July 1st, 2009. We are strongly committed to engage in becoming a Project-Team in 2010.

Strong collaboration with the "Politehnica" University of Bucharest. We have strengthened our collaboration with the *Distributed Systems and Grids* Group of the Computer Science and Engineering Department (http://csite.cs.pub.ro/index.php/en/home) of the *Politehnica University of Bucharest* (PUB). Three of our current PhD students originate from this department. In 2009, we hosted one PhD student (one month) and one Master student (5 months) from PUB, who contributed to a bilateral project under way. An Associate Team Abroad proposal has been accepted in Autumn 2009 in partnership with the MYRIADS Project-Team.

Habilitation and PhD Theses. G. Antoniu defended his Habilitation Thesis (HDR) on March 5, 2009. L. Cudennec defended his PhD Thesis on January 15, 2009.

Marie-Curie European Project. G. Antoniu coordinates the involvement of the INRIA RENNES – BRETAGNE ATLANTIQUE Research Center in the SCALUS project of the Marie-Curie Initial Training Networks programme (ITN), call FP7-PEOPLE-ITN-2008 (2009-2012). Teams involved: KerData and MYRIADS.

Scientific visibility. The KerData Team is a founding member of the CoreGRID ERCIM Working Group on Grids, P2P and Service (http://www.coregrid.net/mambo/content/view/747/418/), led by Frédéric Desprez, GRAAL Project-Team in Lyon.

Software. The BlobSeer open-source software has become publicly available on the INRIA Forge.

3. Scientific Foundations

3.1. Introduction

Managing data at large scales is paramount nowadays. Governmental and commercial statistics, climate modeling, cosmology, genetics, bio-informatics, etc. are just a few examples of fields routinely generating huge amounts of data. It becomes crucial to efficiently manipulate these data, which are typically shared at a global scale. In such a context, one important goal is to provide mechanisms allowing to manage massive data blocks (e.g., of several terabytes), while providing efficient ultra-fine-grain access to literally *microscopic* parts of the data. Several application areas exhibit such a need for efficient scaling to huge data sizes: data mining applications [45], multimedia applications [35], database-oriented applications [39], [60], [53], etc.

.

3.2. Transparent, distributed data sharing

The management of massive data blocks naturally requires the use of data fragmentation and of distributed storage. Grid infrastructures, typically built by aggregating distributed resources that may belong to different administration domains, were built during the last years with the goal of providing an appropriate solution. When considering the existing approaches to grid data management, we can notice that most of them heavily rely on *explicit* data localization and on *explicit* transfers of large amounts of data across the distributed architecture: GridFTP [28], Reptor [48], Optor [48], LDR [18], Chirp [17], IBP [31], NeST [32], etc. Managing huge amounts of data in such an explicit way at a very large scale makes the design of grid application much more complex. One key issue to be addressed is therefore the *transparency* with respect to data localization and data movements. Such a transparency is highly suitable, as it liberates the user from the need to handle data localization and transfers.

Some approaches to grid data management already acknowledge that providing a transparent data access model is important. They integrate this idea at the early stages of their design. *Grid file systems*, for instance, provide a familiar, file-oriented API allowing to transparently access physically distributed data through globally unique, logical file paths. The applications simply open and access such files as if they were stored on a local file system. A very large distributed storage space is thus made available to those existing applications that usually use file storage, with no need for modifications. This approach has been taken by a few projects like GFarm [59], GridNFS [43], LegionFS [63], etc.

On the other hand, the transparent data access model is equally defended by the concept of *grid data-sharing service* [29], illustrated by the JuxMem platform [30]. Such a service provides the grid applications with the abstraction of a globally shared memory, in which data can be easily stored and accessed through global identifiers. To meet this goal, the design of JuxMem leverages the strengths of several building blocks: consistency protocols inspired by Distributed Shared Memory (DSM) systems; algorithms for fault-tolerant distributed systems; protocols for scalability and volatility support from peer-to-peer (P2P) systems. Note that such a system is fundamentally different from traditional DSM systems (such as TreadMarks, etc.). First, it targets a much larger scale through hierarchical consistency protocols suitable for an efficient exploitation of grids made of a federation of clusters. Second, it addresses from the very beginning the problem of resource volatility due to failures or to the lack of resource availability. Compared to the grid file system approach, this approach improves *access efficiency* by totally relying on main memory storage. Besides the fact that a main memory access is more efficient than a disk access, the system can leverage locality-optimization schemes developed for the DSM consistency protocols.

3.3. Managing massive unstructured data under heavy concurrency on large-scale distributed infrastructures

3.3.1. Massive unstructured data: BLOBs

Studies show more than 80% [42] of data globally in circulation is unstructured. On the other hand, data sizes increase at a dramatic level: for example, medical experiments [56] have an average requirement of 1 TB per week. Large repositories for data analysis programs, data streams generated and updated by continuously running applications, data archives are just a few examples of contexts where unstructured data that easily reaches the order of 1 TB. Such unstructured data are often stored as *binary large objects (BLOBs)* within databases or files. However, traditional databases or file systems can hardly cope with BLOBS which grow to huge sizes.

3.3.2. Scalable processing of massive data: heavy access concurrency

To address the scalability issue, specialized programming frameworks like Map-Reduce [37] and Pig-Latin [54] propose high-level data processing frameworks intended to hide the details of parallelization from the user. Such platforms are implemented on top of huge object storage and target high performance by optimizing the

parallel execution of the computation. This leads to *heavy access concurrency* to the BLOBs, thus the need for the storage layer to offer specific support. Parallel and distributed file systems also consider using objects for low-level storage (see next subsection [38], [62], [41]). In other application areas, huge BLOBs need to be used concurrently in the highest layers of applications directly: high-energy physics, multimedia processing [35] or astronomy.

3.3.3. Versioning

When addressing the problem of storing and efficiently accessing very large unstructured data objects [50], [56] in a distributed environment, a challenging case is the one where data is *mutable* and potentially accessed by a very large number of concurrent, distributed processes. In this context, *versioning* is an important feature. Not only it allows to roll back data changes when desired, but it also enables cheap branching (possibly recursively): the same computation may proceed independently on different versions of the BLOB. Versioning should obviously not significantly impact access performance to the object, given that objects are under constant heavy access concurrency. On the other hand, versioning leads to increased storage space usage and becomes a major concern when the data size itself is huge. Versioning efficiency thus refers to both access performance under heavy load and reasonably acceptable overhead of storage space.

3.4. Towards scalable, BLOB-based distributed file systems

Recent research [40] emphasizes a clear move currently in progress from a block-based interface to a object-based interface in storage architectures, with the goal of enabling scalable, self-managed storage networks. It is done by moving low-level functionalities such as space management to storage devices or to storage server, accessed through a standard object interface. This move has a direct impact on the design of today's distributed file systems: object-based file system would then store data rather as objects than as unstructured data blocks. According to [40], this move may eliminate nearly 90% of management workload which was the major obstacle limiting file systems' scalability and performance.

Two approaches exploit this idea. In the first approach, the data objects are stored and manipulated directly by a new type of storage device called *object-based storage device* (OSD). This approach requires an evolution of the hardware, in order to allow high-level object operations to be delegated to the storage device. The standard OSD interface was defined in the Storage Networking Industry Association (SNIA) OSD working group. The protocol is embodied over SCSI and defines a new set of SCSI commands. Recently, a second generation of the command set, Object-Based Storage Devices - 2 (OSD-2) has been defined. The distributed file systems taking the OSD approach assume the presence of such an OSD in the near future and currently rely on a software module simulating its behavior. Examples of parallel/distributed file systems following this approach are Lustre [57] and Ceph [62]. Recently, research efforts [38] have explored the feasibility and the possible benefits of integrating OSDs into parallel file systems, such as PVFS [34].

The second approach does not rely on the presence of OSDs, but still tries to benefit from an object-based approach to improve performance and scalability: files are structured as a set of objects that are stored on storage servers. Google File System [41], and HDFS (Hadoop File System [23]) illustrate this approach.

3.5. Emerging large-scale infrastructures for distributed applications

During the last few years, research and development in the area of large-scale distributed computing led to the clear emergence of several types of physical execution infrastructures for large-scale distributed applications.

3.5.1. Cloud computing infrastructures

The cloud computing model [61], [49], [33] is gaining serious interest from both industry and academia in the area of large-scale distributed computing. It provides a new paradigm for managing computing resources: instead of buying and managing hardware, users rent virtual machines and storage space. Various cloud software stacks have been proposed by leading industry companies, like Google, Amazon or Yahoo!. They aim at providing fully configurable virtual machines or virtual storage (*IaaS: Infrastructure-as-a-Service* [19], [26], [20]), higher-level services including programming environments such as Map-Reduce [37] (*PaaS:*

Platform-as-a-Service [21], [24]) or community-specific applications (*SaaS: Software-as-a-Service* [22], [25]). On the academic side, one of the most visible projects in this area is Nimbus [26], [46], from the Argonne National Lab (USA), which aims at providing a reference implementation for a IaaS. In parallel to these trends, other research efforts focused on the concept of grid operating system: a distributed operating system for large-scale wide-area dynamic infrastructure spanning multiple administrative domains. XtreemOS [52], [27] is such a grid operating system, which provides native support for virtual organizations. Since both the cloud approach and the grid operating system approach deal with resource management on large-scale distributed infrastructures, the relative positioning of these two approaches with respect to each other is currently subject to on-going investigation within the PARIS /MYRIADS Project-Team (http://www.irisa.fr/paris/) at INRIA RENNES – BRETAGNE ATLANTIQUE [51].

In the context of the emerging cloud infrastructures, some of the most critical open issues relate to data management. Providing the users with the possibility to store and process data on externalized, virtual resources from the cloud requires simultaneously investigating important aspects related to security, efficiency and quality of service. Exploring ways to address the main challenges raised by data storage and management on cloud infrastructures is the major factor that motivated the creation of the KerData Research Team (http://www.irisa.fr/kerdata/) of INRIA RENNES – BRETAGNE ATLANTIQUE. To this purpose, it clearly becomes necessary to create mechanisms able to provide feedback about the state of the storage system along with the underlying physical infrastructure. The monitored information can be further fed back into the storage system and used by self-managing engines, in order to enable an autonomic behavior [47], [55], [44], possibly with several goals such as self-configuration, self-optimization, or self-healing.

3.5.2. Petascale infrastructures

In 2011, a new NSF-funded petascale computing system, Blue Waters, will go online at the University of Illinois. Blue Waters is expected to be the most powerful supercomputer in the world for open scientific research when it comes online. It will be the first system of its kind to sustain one petaflop performance on a range of science and engineering applications. The goal of this facility is to open up new possibilities in science and engineering by providing computational capability that makes it possible for investigators to tackle much larger and more complex research challenges across a wide spectrum of domains: predict the behavior of complex biological systems, understand how the cosmos evolved after the Big Bang, design new materials at the atomic level, predict the behavior of hurricanes and tornadoes, and simulate complex engineered systems like the power distribution system and airplanes and automobiles.

To reach sustained-petascale performance, Blue Waters relies on advanced, dedicated technologies under development at IBM at several levels: processor, memory subsystem, interconnect, operating system, programming environment, system administration tools. A similar effort was initiated by RIKEN (Japan), who aimed to build a next-generation supercomputer targeting 10 Petaflops performance in its research center in Kobe. (This program was stopped by the Japanese government, however this decision may be reconsidered.)

In the context of such efforts whose goal is to provide sustained Petascale (and beyond Petascale) performance, data management is again a critical issue that highly impacts the application behavior. Petascale supercomputers exhibit specific architectural features (e.g., a multi-level memory hierarchy scalable to tens to hundreds of thousands of nodes). It needs to be specifically taken into account in order to enable a parallel file system to fully benefit from the capabilities of the machine. Providing scalable data throughput on such unprecedented scales is clearly an open challenge today.

3.5.3. Desktop grids

During the recent years, Desktop grids have been extensively investigated as an efficient way to build cheap, large-scale virtual supercomputers by gathering idle resources from a very large number of users. Physical infrastructures for Grid Computing typically rely on clusters of workstations belonging to institutions, and interconnected through dedicated, high-throughput wide-area networks. In contrast, desktop grids rely on individual desktop computers, interconnected through Internet, provided by volunteer users. The initial, widely-spread usage of Desktop grids for parallel applications consisting in non-communicating tasks with small input/output parameters is a direct consequence of the physical infrastructure (volatile nodes, low bandwidth),

unsuitable for communication-intensive parallel applications with high input or output requirements. However, the increasing popularity of volunteer computing projects has progressively lead to attempts to enlarge the set of application classes that might benefit of Desktop Grid infrastructures. If we consider distributed applications where tasks need very large input data, it is no longer feasible to rely on classical centralized server-based Desktop Grid architectures, where the input data was typically embedded in the job description and sent to workers: such a strategy could lead to significant bottlenecks as the central server gets overwhelmed by download requests. To cope with such data-intensive applications, alternative approaches have been proposed, with the goal of offloading the transfer of the input data from the central servers to the other nodes participating to the system, with potentially under-used bandwidth.

Two approaches follow this idea. One of them adopts a P2P strategy, where the input data gets spread across the distributed Desktop Grid (on the same physical resources that serve as workers) [36]. A central data server is used as an initial data source, from which data is first distributed at a large scale. The workers can then download their input data from each other when needed, using for instance a BitTorrent-like mechanism. An alternative approach [36] proposes to use Content Distribution Networks (CDN) to improve the available download bandwidth by redirecting the requests for input data from the central data server to some appropriate surrogate data server, based on a global scheduling strategy able to take into account criteria such as locality or load balancing. The CDN approach is more costly then the P2P approach (as it relies on a set of data servers), however it is potentially more reliable (as the surrogate data servers are supposed to be stable enough).

More recent research makes a step further and considers using Desktop grids for distributed applications with high *output* data requirements. Each such application consists of a set of distributed tasks that *produce and potentially modify large amounts of data* in parallel, under heavy concurrency conditions. Such characteristics are featured by 3D rendering applications, or massive data processing applications that produce data transformations. Such a context requires new approaches to data management, in order to cope with both input and output data in a scalable way.

3.6. Emerging programming models for scalable data-management

MapReduce is a parallel programming paradigm successfully used by large Internet service providers to perform computations on massive amounts of data. A computation takes a set of input key/value pairs, and produces a set of output key/value pairs. The user of the MapReduce library expresses the computation as two functions: *map*, that processes a key/value pair to generate a set of intermediate key/value pairs, and *reduce*, that merges all intermediate values associated with the same intermediate key. The framework takes care of splitting the input data, scheduling the jobs' component tasks, monitoring them and re-executing the failed ones. After being strongly promoted by Google, it has also been implemented by the open source community through the Hadoop project, maintained by the Apache Foundation, and supported by Yahoo! and even by Google itself. This model is currently getting more and more popular as a solution for rapid implementation of distributed data-intensive applications. The key strength of the Map/Reduce model is its inherently high degree of potential parallelism that should enable processing of petabytes of data in a couple of hours on large clusters consisting of several thousands of nodes.

At the core of the Map/Reduce frameworks stays a key component: the storage layer. To enable massively parallel data processing to a high degree over a large number of nodes, the storage layer must meet a series of specific requirements. Firstly, since data is stored in huge files, the computation will have to efficiently process small parts of these huge files concurrently. Thus, the storage layer is expected to provide efficient *fine-grain access* to the files. Secondly, the storage layer must be able to sustain a *high throughput* in spite of *heavy access concurrency* to the same file, as thousands of clients simultaneously access data.

These critical needs of data-intensive distributed applications have not been addressed by classical, POSIX-compliant distributed file systems. Therefore, specialized file systems have been designed, such as HDFS, the default storage layer of Hadoop. HDFS has however some difficulties in sustaining a high throughput in the case of concurrent accesses to the same file. Amazon's cloud computing initiative, Elastic MapReduce, employs Hadoop on their Elastic Compute Cloud infrastructure (EC2) and inherits these limitations. The storage backend used by Hadoop is Amazon's Simple Storage Service (S3), which provides limited support

for concurrent accesses to shared data. Moreover, many desirable features are missing altogether, such as the support for versioning and for concurrent updates to the same file. Finally, another important requirement for the storage layer is its ability to expose an interface that enables the application to be *data-location aware*. This is critical in order to allow the scheduler to use this information to place computation tasks close to the data and thus reduce network traffic, contributing to a better global data throughput.

4. Application Domains

4.1. Application Domains

The research carried out within the KerData Team targets the following classes of applications.

- Grid and cloud data-mining applications handling massive data distributed at a large scale (e.g. through MapReduce data analysis).
- Advanced data services for cloud infrastructures requiring efficient data sharing under heavy concurrency.
- Distributed storage for Petaflop computing applications.
- Data storage for desktop grid applications with high write throughput requirements.
- Distributed data sharing and storage for extremely large databases.

5. Software

5.1. JuxMem

Participants: Gabriel Antoniu, Luc Bougé, Loïc Cudennec.

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

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

License: GNU Lesser General Public License (LGPL) version 2.1.

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

Presentation: JUXMEM is a supportive platform for a data-sharing service for grid computing. This service addresses the problem of managing mutable data on dynamic, large-scale configurations. It can be seen as a hybrid system combining the benefits of *Distributed Shared Memory* (DSM) systems (transparent access to data, consistency protocols) and *Peer-to-Peer* (P2P) systems (high scalability, support for resource volatility). JUXMEM's architecture decouples fault-tolerance management from consistency management. Multiple consistency protocols can be built using fault-tolerant building blocks such as *consensus*, *atomic multicast*, *group membership*. Currently, a hierarchical protocol implementing the entry consistency model is available. A more relaxed consistency protocol adapted to visualization is also available. Up to version 0.4 (included), JuxMem is based on the *JXTA* generic platform for P2P services (Sun Microsystems, http://www.jxta.org/). This version includes 16,700 lines of Java code and 16,000 lines of C code. Implementation started in February 2003. In 2008, a lighter version of JuxMem (0.5), not dependent on JXTA was released. It includes 4600 lines of C++ code. JUXMEM has been used for transparent data sharing within the following ANR projects: ANR CI LEGO (ended in June 2009), and ANR MD RESPIRE (ended in December 2008). An industrial collaboration with Sun Microsystems centered on JUXMEM funded Loïc Cudennec's PhD thesis, defended in January 2009. JUXMEM is currently used within an international collaboration with the University of Tsukuba. Other past users: University of Illinois at Urbana Champaign, University of Pisa, University of Calabria.

5.2. CoRDAGe

Participants: Gabriel Antoniu, Luc Bougé, Loïc Cudennec.

Contact: Loïc Cudennec, `Loic.Cudennec@cea.fr`

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

License: GNU Lesser General Public License (LGPL) version 3.

Status: Registered at the *Agence pour la Protection des Programmes* (APP) under the Inter Deposit Digital Number `IDDN.FR.001.090003.000.S.P.2009.000.10000`.

Presentation: CoRDAGe is a generic co-deployment and re-deployment service for grid computing applications. It addresses the deployment of applications in a dynamic way: it allows redeployment and reconfiguration during the execution, as well as coordinated deployment of multiple, heterogeneous applications. The service interfaces applications with grid reservation and deployment tools, thus making all interactions transparent for the applications and the final user. CoRDAGe has been developed since January 2008 within an INRIA Gforge project. The current implementation features near all functionalities that come with the model. It includes more than 6,700 lines of C++, C and Perl code. It relies on the ADAGE deployment tool and the *OAR* resource scheduler. CoRDAGe can handle applications based on the *JXTA* peer-to-peer system, the JUXMEM data-sharing service and the *Gfarm* distributed file system. It has been tested within the GRID'5000 experimental platform, using up to 386 nodes and 6 sites in a single experiment. The CoRDAGe prototype has been used within the ANR CI LEGO project (ended in June 2009), and within the ANR MD RESPIRE project (ended in December 2008).

5.3. BlobSeer

Participants: Gabriel Antoniu, Luc Bougé, Bogdan Nicolae.

Contact: Bogdan Nicolae, `Bogdan.Nicolae@irisa.fr`

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

License: GNU Lesser General Public License (LGPL) version 3.

Status: This software is available on INRIA's forge. Registration with APP is in progress.

Presentation: BlobSeer is a data storage service specifically designed to deal with the requirements of large-scale data-intensive distributed applications, that abstract data as huge sequences of bytes which are stored as BLOBs (binary large objects). It exports a simple, yet versatile versioning interface to manipulate BLOBs that enables reading, writing and appending to them. BlobSeer offers both scalability and performance with respect to a series of issues typically associated with the data-intensive context: *scalable aggregation of storage space* from the participating nodes with minimal overhead, ability to store *huge data objects*, *efficient fine-grain access* to data subsets, *high throughput in spite of heavy access concurrency*, as well as *fault-tolerance*. Development has started since January 2008. The implementation is build on top of the Boost collection of C++ libraries, Berkeley DB and libconfig. Additional scripting in Perl/Python handles deployment on GRID'5000, which is done through the *OAR* resource scheduler. Benchmarking so far has proved correctness and performance with up to 400 nodes from 3 different sites.

6. New Results

6.1. BlobSeer

Participants: Bogdan Nicolae, Gabriel Antoniu, Luc Bougé.

Starting from a preliminary experimental implementation, we developed BlobSeer to a fully fledged data-storage service for large-scale distributed data-intensive applications that process unstructured data, which is stored as *huge sequences of bytes: BLOBs*. We focused on demonstrating the benefits of using versioning when manipulating such large sequences of bytes, as well as the benefits of using data and metadata decentralization to support heavy write access concurrency efficiently.

6.1.1. Efficient Versioning for Large Object Storage

We targeted large-scale data-intensive distributed applications built on top of paradigms that exploit data parallelism explicitly. In this context, applications need to acquire and maintain huge unstructured datasets, while performing computations in the background over these datasets.

We formalized a simple, yet versatile versioning-oriented access interface to optimize the data management. This interface enables creating a BLOB, reading/writing parts of the BLOB and appending new data to the BLOB. Data is never overwritten as each time a write/append occurs, a new snapshot of the BLOB is created. Read operations are forced to access a particular snapshot explicitly, thus enabling readers to be decoupled from writers and thus allowing data gathering and data processing to avoid the need of synchronizing between each other. Moreover, we guarantee linearizability for all operations, thus eliminating the need of explicit synchronization at operation level. Finally, we illustrated the benefits of using our interface in a real-life, data-intensive MapReduce scenario.

As a next step, we extended BlobSeer to provide efficient support for the interface we proposed. This involved implementing the append operation (missing from our previous implementation) and further develop our distributed metadata management scheme to accommodate this operation efficiently while maintaining the same level of performance for reads and writes.

We conducted preliminary large-scale experimentation on the Grid'5000 testbed evaluating append performance. Results suggest a good scalability with respect to the data size and to the number of concurrent accesses. There results have been published in [8].

6.1.2. High Write Throughput in Desktop Grids

We evaluated BlobSeer in its role as a storage service for *write-intensive* applications running in Desktop Grids that have high *output* data requirements and where the access grain and the access pattern may be random.

In this context, the main challenge is to deal with *heavy write concurrency* in an efficient way. We addressed this challenge by combining data striping with our decentralized, versioning-oriented metadata structure built on top of distributed segment trees and spread over a Distributed Hash Table (DHT).

To prove the benefits of our decentralized approach to data and metadata management, we conducted extensive experimentation on the Grid'5000 testbed. We evaluated both the impact of data decentralization and metadata decentralization. In a final large-scale experiment, we demonstrated the importance of the latter on sustaining high write throughput under heavy write concurrency. The results suggest clear benefits of using a decentralized metadata approach. They have been published in [9].

6.2. Map-Reduce

Participants: Diana Moise, Bogdan Nicolae, Gabriel Antoniu, Luc Bougé, Matthieu Dorier.

We focused on improving the MapReduce framework, by enhancing it with features provided by a data-storage service such as BlobSeer. We started by analyzing a key component of MapReduce frameworks, the storage layer. To enable massively parallel data processing over a large number of nodes, the storage layer must meet a series of specific requirements, that standard distributed file systems do not include in their design. Firstly, since data is stored in huge files, the computation has to process small parts of these huge files concurrently. Thus, the storage layer is expected to provide efficient *fine-grain access* to the files. Secondly, the storage layer must be able to sustain a *high throughput* in spite of *heavy access concurrency* to the same file, as thousands of clients may simultaneously access data. *Versioning* in this context becomes an important feature that is expected from the storage layer. Not only it enables rolling back undesired changes, but also branching a

dataset into two independent datasets that can evolve independently. Finally, another important requirement for the storage layer is its ability to expose an interface that enables the application to be *data-location aware*. The scheduler uses this information to place computation tasks close to the data, thus reducing the network traffic, and contributing to a better global data throughput.

As BlobSeer already provides these features, the next step was to add a layer that enabled it to be used as a file system on top of BlobSeer. We called this additional layer, the BlobSeer *File System* (BSFS). This layer consists in a centralized *namespace manager*, which is responsible for maintaining a file system namespace, and for mapping files to BLOBs. We also implemented a caching mechanism for read/write operations, as MapReduce applications usually process data in small records (4 KB). This mechanism prefetches a whole block when the requested data is not already cached, and delays committing writes until a whole block has been filled in the cache. To make the MapReduce scheduler data-location aware, we extended BlobSeer with a new primitive, that exposes the block allocation to providers.

To evaluate the benefits of using BlobSeer as the storage backend for MapReduce applications we used Hadoop - Yahoo!'s implementation of the MapReduce framework. We substituted the original data storage layer of Hadoop (the *Hadoop Distributed File System - HDFS* with our BlobSeer-based file system - BSFS. To measure the impact of our approach, we performed experiments both with synthetic microbenchmarks and real MapReduce applications. The experiments were conducted on the Grid'5000 testbed, using up to 270 nodes. We focused on scenarios that exhibit highly concurrent accesses to shared files. The results showed that Hadoop significantly improved its sustained throughput by using BSFS instead of its default storage layer. These results will be presented at the 2010 IPDPS Conference [15].

6.3. Introspective BlobSeer

Participants: Alexandra Carpen-Amarie, Jing (Tylor) Cai, Alexandru Costan, Gabriel Antoniu, Luc Bougé.

The cloud computing model is an emerging paradigm for dynamically provisioning processing time and storage space from a cloud of computational resources. The most important layer in the cloud-computing stack is the *Infrastructure-as-a-Service* (IaaS), which provides fully-configurable virtual machines or virtual storage. In the context of the emerging cloud infrastructures, one of the most critical challenges concerns data management. Our work focuses on building an autonomic, efficient and secure storage service for IaaS clouds, designed to leverage the needs of data-intensive distributed applications by leveraging BlobSeer, the large-scale distributed data-sharing platform developed in our team.

The first step towards an autonomic data-sharing system was to equip the BlobSeer platform with introspection capabilities. This feature plays a crucial role in helping the users to overcome the issues raised by managing the behavior of their systems at large scales. Our work addressed the challenges raised by the introduction of introspection into such a data-management system. These challenges come from the fact that introspection is often limited to low-level tools for monitoring the physical nodes, whereas enabling an autonomic behavior for our system requires the analysis of both general and specific data-storage parameters, such as physical data distribution or data access patterns.

We proposed a 3-layered architecture [13] built on top of BlobSeer: 1) an instrumentation layer that extracts the low-level, raw data from the different components of BlobSeer; 2) a monitoring layer that deals with collecting and storing the monitoring data from the instrumentation layer; and 3) an introspective layer that processes the gathered data into higher-level information describing the state and the behavior of the system. The data extracted by the introspective layer that can be further fed to a self-adaptive engine, able to improve the performance and to optimize the resource usage in BlobSeer.

The monitoring layer was implemented as an extension [11] of a general-purpose, large-scale monitoring framework, called MonALISA. The proposed architecture was evaluated on the Grid'5000 testbed, using more than 100 nodes for the experiments. The performed experiments confirm the outcome of the introspection layer, by means of graphical representations associated with the various high-level data extracted [12].

We are now investigating several directions that will lead to the integration of the BlobSeer platform within an IaaS cloud, as a storage service. One direction, which builds upon the instrumentation capabilities developed so far, is the design and integration of the self-adaptation layer which will enable an autonomic behavior.

The second direction is related to security issues raised by the design of BlobSeer, which need to be addressed when exposing BlobSeer as a service for sharing data belonging to different users. We focused on the detection of the illegal actions performed by malicious clients, by relying on the previously-designed introspection architecture. The same framework can be further extended to enforce restrictions on the client actions, in order to cope with clients breaking access policies or with abnormal client activity. Another important direction we are currently exploring is integrating BlobSeer with an existing cloud infrastructure, such as the Nimbus cloud environment from Argonne National Lab.

6.4. Towards a BLOB-based file system

Participants: Viet-Trung Tran, Gabriel Antoniu, Bogdan Nicolae.

Most object-based file systems exhibit a decoupled architecture that generally consists of two layers: a low-level object management service, and a high-level file system metadata management. We have explored how this two-layer approach could be used in order to build an object-based grid file system for applications that need to manipulate huge data, distributed and concurrently accessed at a very large scale. We have investigated this approach by experimenting how the Gfarm grid file system developed at the University of Tsukuba could leverage the properties of the BlobSeer distributed object management service, specifically designed for huge data management under heavy concurrency.

We thus leverage Gfarm's powerful file metadata capabilities and rely on BlobSeer for efficient and transparent low-level distributed object storage. The goal is to build a BLOB-based grid file system that exhibits scalable file access performance in scenarios where huge files are subject to massive, concurrent, fine-grain accesses. Instead of using the local disk for data storage, each Gfarm storage node stores data in BlobSeer. The benefits are mutual: by delegating object management to BlobSeer, Gfarm can expose efficient fine-grain access to huge files and benefit from transparent file striping (TB size). On the other hand, BlobSeer benefits from the file system interface on top of its current API. We have defined and implemented a preliminary integrated architecture, and we have evaluated it through a series of preliminary experiments conducted on the Grid'5000 testbed. The resulting BLOB-based grid file system exhibits scalable file access performance in scenarios where huge files are subject to massive, concurrent, fine-grain accesses. These results have been published in [10].

6.5. Improving QoS in Large-scale Distributed Data Storage Services

Participants: Bogdan Nicolae, Jesús Montes, Gabriel Antoniu.

The ability to sustain a stable high throughput for data access is a highly desirable property for large scale distributed storage systems, as it strongly impacts the quality of service offered by the storage system and thereby the overall performance of applications running on top of the storage service. Handling quality of service in a large-scale distributed system is however a very difficult task, as a very large number of factors are involved: the data access patterns, the status of a huge number of physical components, etc. Thus, conventional profiling and analysis is of little use.

We proposed an offline analysis approach to improve the quality of service in distributed storage systems based on global behavior modeling combined with client-side quality of service feedback. It automates the process of identifying dangerous behavior patterns in storage services, which makes reasoning about potential improvements much easier.

We demonstrated our approach, by applying GloBeM, a global behavior modeling technique based on monitoring data analysis and machine learning, to improve the quality of service in BlobSeer. We evaluated the improvement through extensive experiments on the Grid'5000 testbed under hard conditions: highly-concurrent data access patterns, for long periods of service uptime, while supporting failures of the physical storage components.

Our results show substantial improvement in sustaining a higher and more stable data access throughput. They have been submitted for publication [16].

6.6. Distributed random number generator

Participants: Benjamin Girault, Bogdan Nicolae, Luc Bougé, Gabriel Antoniu.

This work has been carried out by Benjamin Girault, student at ENS CACHAN, during its *Magistère* Research Intership, June-July 2009, under the supervision of Bogdan Nicolae, Luc Bougé and Gabriel Antoniu.

The objective was to provide a convenient platform to design a distributed Random Number Generator (RNG). A large number of copies of classical RNGs were run in parallel on the nodes of the GRID'5000 platform. Their outputs are combined into a very large string of numbers using BlobSeer to manage these highly-concurrent accesses to a shared data. This resulted in both high throughput and high unpredictability.

These simulations on GRID'5000 involved up to 415 nodes during a whole week-end, to gather enough data from slow RNGs to analyze them. The volume of data gathered during this week-end was of about 7 MB. For faster RNGs, the program ran until that 16GB of random numbers were gathered, which is the amount needed to efficiently compare RNGs.

Comparing various RNGs showed that the generator which gave the best results in this context is the one based on HAVEGE [58], developed at INRIA RENNES – BRETAGNE ATLANTIQUE by André Seznec and Nicolas Sendrier.

A description of the experiment together with a statistical evaluation of the RNG quality can be found in the internship report https://gforge.inria.fr/docman/view.php/217/6485/main.pdf

This work is part of a wider goal which is to create tools to generate, test and evaluate RNGs in a distributed environment, led by Alin Suciu's team at the Faculty of Automation and Computer Science Computer Science, Technical University of Cluj-Napoca (TUCN), Romania.

7. Other Grants and Activities

7.1. Local intiatives

Participant: Gabriel Antoniu.

The UNIVERSITY RENNES 1 has allocated specific funding to Action *Roumanie - systèmes numériques* (3000 Euros), coordinated by Gabriel Antoniu. This action aims at establishing and intensifying scientific exchanges between Romanian Universities and teams from IRISA. Most exchanges took place with Team *Distributed Computing and Grids* from the "Politehnica" University of Bucharest (PUB), Computer Science Department. This funding has been used to co-fund the internship of Mihaela Vlad, MS student from PUB, hosted by the KerData Team through INRIA's Internship's Programme, which covers 50% of the costs of her stay).

7.2. Regional intiatives

7.2.1. PhD grant

Participant: Diana Moise.

The Brittany Regional Council provides half of the financial support for the PhD thesis of Diana Moise (GRID5000BD project). This support amounts to a total of around 14,000 Euros/year.

7.3. National initiatives

7.3.1. ANR CI LEGO Project

Participants: Gabriel Antoniu, Loïc Cudennec.

The aim of this project was to provide algorithmic and software solutions for large-scale architectures, focusing on performance issues. The project addresses topics in programming models, data management, communication models, and scheduling. The results have been validated on three applications: an ocean-atmosphere numerical simulation, a cosmology simulation, and a sparse-matrix solver. This is a project which started in January 2006 for three years and was granted a no-cost 6-month extension in 2009. Project site: http://graal.ens-lyon.fr/LEGO/. In 2009, we contributed to the final project demonstrator: we focused on providing deployment support for all project components through the CoRDaGE deployment tool developed within the framework of Loïc Cudennec's PhD thesis.

7.4. European intiatives

7.4.1. SCALUS: Marie-Curie Initial Training Network (FP7)

Participants: Gabriel Antoniu, Luc Bougé, Bogdan Nicolae, Alexandra Carpen-Amarie, Diana Moise, Viet-Trung Tran.

The consortium of the SCALUS Marie-Curie Initial Training Network (MCITN) project aims at elevating education, research, and development inside the area of large-scale, distributed ubiquitous storage with a focus on cluster, grid, and cloud storage. The vision of this MCITN is to deliver the foundation for ubiquitous storage systems, which can be scaled in arbitrary directions (capacity, performance, distance, security, . . .). The consortium's goal is to build the first interdisciplinary teaching and research network on storage issues. It consists of top European institutes and companies in storage and cluster technology, building a demanding but rewarding interdisciplinary environment for young researchers. This interdisciplinary research consortium is the foundation for young researchers to be able to perform the innovative research tasks outlined in this proposal. The academic partners include INRIA RENNES – BRETAGNE ATLANTIQUE, Universidad Politécnica de Madrid, Barcelona Supercomputing Center, University of Paderborn, Ruprecht-Karls-Universität Heidelberg, Durham University, FORTH, École des Mines de Nantes, XLAB, CERN, NEC, Microsoft Research, Fujitsu, Sun Microsystems. The project has started on December 1st, 2009, for 4 years. It involves the KerData and MYRIADS teams. Gabriel Antoniu serves as a coordinator for INRIA RENNES – BRETAGNE ATLANTIQUE.

7.4.2. GridDataViz: CNRS-Romanian Science Academy Cooperation Programme

Participants: Gabriel Antoniu, Bogdan Nicolae, Alexandra Carpen-Amarie, Diana Moise.

This is a bilateral contract with the *Distributed Systems and Grids* Group lead by Nicolae Tapus and Valentin Cristea at "Politehnica" University of Bucharest, Romania. This project was accepted in the framework of a call for projects jointly issued by the French CNRS and by the Romanian Academy of Science for Period January 2008 – December 2009. It focused on visualization and remote control of distributed grid data sharing platforms based on P2P techniques. This project is part of a larger collaboration between the KerData team and "Politehnica" University of Bucharest.

In 2009, a PhD student from PUB was hosted by the KerData Team: Alexandru Costan (one month, funded by the bilateral CNRS-Romanian Academy project). Additionally, an MS student from PUB was hosted this year for 5 months for her Master research project : Mihaela Vlad (co-funded by INRIA's Internships programme and by the UNIVERSITY RENNES 1, see above). This project led to a preliminary definition of an introspection layer for the BlobSeer data-management service based on the MonALISA distributed monitoring framework, see [13]. It serves as a starting point for the proposal of an INRIA Associate Team to start in January 2010.

7.4.3. GridRand: Bilateral PHC contract with the Technical University of Cluj-Napoca, Romania

Participants: Gabriel Antoniu, Bogdan Nicolae.

GridRand is joint PHC project of INRIA RENNES – BRETAGNE ATLANTIQUE and the Technical University of Cluj-Napoca, Romania. It is an interdisciplinary research project, of both theoretical and practical nature, situated at the confluence of two main research areas: the first area is Generation and Testing of Random Number Sequences (RNS), a well-established research area. It started by Shannon's seminal work on information theory and followed by contributions from Von Neumann, Knuth, Kolmogorov, and many others, with applications in areas like digital and quantum cryptography or Monte Carlo and Quasi-Monte Carlo methods. The second area is grid computing. Our goal is to address the following question: how to generate, test and manage large random sequences efficiently, by making good use of the resources offered by the Grid infrastructure and technologies?

In 2009, Bogdan Nicolae made two one-week visits at the Technical University of Cluj-Napoca and the Romanian partners visited the KerData Team for one week. Benjamin Girault, student at ENS CACHAN was hosted by the KerData Team during its *Magistère* Research Intership, June-July 2009 and worked on this project.

7.5. International intiatives

7.5.1. CNRS-JST project

Participants: Gabriel Antoniu, Loïc Cudennec, Bogdan Nicolae, Viet-Trung Tran.

Masahiro Nakamura, undergraduate student from the University of Tsukuba, contributed to this work.

NEGST (*NExt Grid Systems and Techniques*) is a project targeting international collaboration and promotion on interoperability and advanced grid technologies (http://www2.lifl.fr/MAP/negst/). Within this framework we collaborated with Osamu Tatebe from the University of Tsukuba in the area of large-scale data-sharing.

In 2009, the KerData Team hosted Masahiro Nakamura, engineering student at the University of Tsukuba, for a 3-week internship. He performed a preliminary experimental study that was continued by Viet-Trung Tran during his Master research internship. This work (published as [10]) aimed at investigating approaches to integrate the Gfarm grid file system developed at the University of Tsukuba with the BlobSeer BLOB-based data management system. The goal is to build an object-based grid file system for applications that need to manipulate huge data, distributed and concurrently accessed at a very large-scale. Viet-Trung Tran will continue to explore this subject during his thesis, started in October 2009. This work will continue in partnership with Osamu Tatebe (University of Tsukuba), with whom a bilateral PHC Sakura project has been submitted in September 2009. Finally, KerData will participate to larger project (a follow-up of the NEGST project) that will be submitted to a joint ANR-JST call.

7.6. Other contacts

7.6.1. Kate Keahey, Argonne National Laboratory, USA

We had informal contacts with Kate Keahey (http://www.mcs.anl.gov/~keahey/) from Argonne National Laboratory, USA, leader of the Nimbus cloud computing project. Gabriel Antoniu visited Kate's team at ANL in December 2009. There, he gave a seminar and had discussions about collaboration opportunities focusing on the usage of BlobSeer in Nimbus-enabled cloud infrastructures. In this context, Kate Keahey's team at ANL will host Bogdan Nicolae for a 3-month doctoral stay.

7.6.2. Marc Snir, National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana Champaign, USA

Gabriel Antoniu was invited to participate to the 2nd workshop of the NCSA-UIUC/INRIA Joint Laboratory for Petascale Computing in December 2009. He took part in exploratory discussions in the area of distributed storage for Petascale architectures, with a specific focus on the Blue Waters machine (http://www.ncsa.illinois.edu/BlueWaters/), expected to become the world's most powerful supercomputer in 2011. Gabriel will visit NCSA-UIUC again in 2010 with Bogdan Nicolae, to explore how the BlobSeer BLOB-based approach developed by KerData could be used in an optimized distributed file system for Blue Waters.

7.6.3. Indranil Gupta, University of Illinois at Urbana Champaign, USA

During his visit at NCSA-UIUC in December 2009, Gabriel Antoniu took contact with Indranil Gupta, co-responsible for the Cloud Compute Testbed, an experimental platform for cloud computing co-funded by NSF through UIUC, in partnership with Yahoo!, Intel and HP. The discussions explored possible collaborations on the execution of MapReduce-based applications on cloud testbeds.

7.6.4. Roberto Baldoni, University of Rome "La sapienza", Italy

We had informal contacts with Roberto Baldoni from the University of Rome (http://www.dis.uniroma1.it/~baldoni/) about the usage of BlobSeer as an efficient data management substrate for MapReduce-based financial applications. Roberto Baldoni served as a referee for Gabriel Antoniu's Habilitation thesis. Gabriel visited Roberto Baldoni's team in June 2009, where he gave a seminar and discussed opportunities for collaboration.

8. Dissemination

8.1. Committees

8.1.1. Leaderships, Steering Committees and community service

Euro-Par Conference Series. L. Bougé serves as a Vice-Chair of the *Steering Committee* of the *Euro-Par* annual conference series on parallel computing.

Agrégation of Mathematics. L. Bougé serves as a Vice-Chair of the National Selection Committee for High-School Mathematics Teachers, Informatics Track.

NAS-2010 Conference. G. Antoniu serves as a Vice-Chair of the *Program Committee* for the storage track of the *IEEE NAS* international conference on Networking, Architecture, and Storage.

SCALUS Marie-Curie Initial Training Networks project. G. Antoniu coordinates the involvement of the INRIA RENNES – BRETAGNE ATLANTIQUE Research Center in the SCALUS project of the Marie-Curie Initial Training Networks programme (ITN), call FP7-PEOPLE-ITN-2008 (2009-2012).

CoreGRID ERCIM Working Group. G. Antoniu coordinates the involvement of the INRIA RENNES – BRETAGNE ATLANTIQUE Research Center in the CoreGRID ERCIM Working Group.

8.1.2. Editorial boards, direction of program committees

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

8.1.3. Program Committees

G. Antoniu served in the Program Committees for the following conferences and workshops: DaMaP 2009, HiperGrid 2009, RenPar'19, MapReduce 2010, ADiS 2010.

L. Bougé served in the Program Committee for the following conferences: NPC 2009.

8.1.4. Evaluation committees, consulting

L. Bougé served as a member of the Selection Committee for the *Gilles Kahn PhD Thesis Award 2009*.

8.2. Research schools

G. Antoniu gave 9 hours lectures on grid data management at the June 2009 CEA-EDF-INRIA Summer School on Emerging Grid Middleware Standards (http://www.inria.fr/actualites/colloques/cea-edf-inria/2009/grid/index.en.html).

8.3. Invited talks

G. Antoniu gave an invited talk entitled *Hierarchical approaches for large-scale data management based on Gfarm, JuxMem and* BlobSeer*: current status and work in progress* at the PAAP France-Japan workshop held in April 2009 in Kyoto, Japan.

G. Antoniu gave an invited talk entitled *Bringing together fault tolerance and data consistency to enable grid data sharing* at the Dagstuhl seminar on Fault Tolerance in High-Performance Computing and Grids held in May 2009 in Dagstuhl, Germany.

G. Antoniu gave an invited talk entitled BlobSeer*: Enabling High Data Throughput under Heavy Access Concurrency Through Decentralized Data and Metadata Management* at the University of Rome "La Sapienza" in June 2009.

G. Antoniu gave an invited talk entitled BlobSeer*: Enabling efficient lock-free, versioning-based storage for massive data under heavy access concurrency* at the 2nd workshop of the Joint Laboratory for Petascale Computing held in December 2009 at NCSA, Urbana, IL, USA.

G. Antoniu gave an invited talk entitled BlobSeer*: efficient cloud storage for massive data under heavy access concurrency* at Argonne National Lab, IL, USA in December 2009.

G. Antoniu was invited to give a keynote talk entitled *Autonomic cloud storage: challenges at stake* at the ADiS workshop held in February 2010 at Krakow, Poland.

8.4. Doctoral teaching

Only the teaching contributions of project-team members on non-teaching positions are mentioned below.

G. Antoniu gave lectures on peer-to-peer systems within the *Peer-to-Peer Systems* Module of the Master Program (2nd year), UNIVERSITY RENNES 1, and within the *Distributed Systems* Module taught for the final year engineering students of INSA Rennes. He gave lectures on Grid Data Management within the *Distributed Architectures* Module of the ALMA Master Program (2nd year) of the University of Nantes. He also taught a full course on *Grid Computing* for final year engineering students at the ESIEA Engineering School, Paris.

8.5. Administrative responsibilities

G. Antoniu serves as the Scientific Correspondent for International Relations of the INRIA RENNES – BRETAGNE ATLANTIQUE Research Center.

L. Bougé chairs the Computer Science and Telecommunication Department (*Département Informatique et Télécommunications, DIT*) of the Brittany Extension of ENS CACHAN. He leads the Master Program (*Magistère*) in Computer Science at the Brittany Extension of ENS CACHAN.

8.6. Miscellaneous

L. Bougé is a member of the Project-Team Committee of IRISA (*Comité des projets*), standing for the ENS CACHAN partner.

9. Bibliography

Major publications by the team in recent years

[1] G. ANTONIU, L. BOUGÉ, M. JAN. *JuxMem: An Adaptive Supportive Platform for Data Sharing on the Grid*, in "Scalable Computing: Practice and Experience", vol. 6, n^{o} 33, 2005, p. 45-55, http://hal.inria.fr/inria-00000984/en/.

[2] G. ANTONIU, L. CUDENNEC, M. GHAREEB, O. TATEBE. *Building Hierarchical Grid Storage Using the Gfarm Global File System and the JuxMem Grid Data-Sharing Service*, in "Euro-Par 2008 Parallel Processing, 14th International Euro-Par Conference, Las Palmas de Gran Canaria, Spain", Lecture Notes in Computer Science, vol. 5168, Springer, University of Las Palmas, 2008, p. 456-465, http://hal.inria.fr/inria-00318590/en/JP.

[3] G. ANTONIU, L. CUDENNEC, M. JAN, M. DUIGOU. *Performance scalability of the JXTA P2P framework*, in "Proc. IEEE International Parallel and Distributed Processing Symposium (IPDPS 2007), Long Beach, USA", 2007, 108, http://hal.inria.fr/inria-00178653/en/US.

[4] G. ANTONIU, J.-F. DEVERGE, S. MONNET. *How to bring together fault tolerance and data consistency to enable grid data sharing*, in "Concurrency and Computation: Practice and Experience", n^{o} 17, 2006, p. 1-19, http://hal.inria.fr/inria-00000987/en/.

[5] L. CUDENNEC, G. ANTONIU, L. BOUGÉ. *CoRDAGe: towards transparent management of interactions between applications and ressources*, in "International Workshop on Scalable Tools for High-End Computing (STHEC 2008), Kos, Greece", 2008, p. 13-24, http://hal.inria.fr/inria-00288339/en/, Held in conjunction with the International Conference on Supercomputing (ICS 2008).

Year Publications

Doctoral Dissertations and Habilitation Theses

[6] G. ANTONIU. *Contribution à la conception de services de partage de données pour les grilles de calcul*, École Normale Supérieure de Cachan - Antenne de Bretagne, March 2009, http://tel.archives-ouvertes.fr/tel-00437324/fr/, Habilitation à Diriger des Recherches (Habilitation Thesis, HDR).

[7] L. CUDENNEC. *CoRDAGe : Un service générique de co-déploiement et redéploiement d'applications sur grilles*, University Rennes 1, January 2009, http://tel.archives-ouvertes.fr/tel-00357473/en/, Ph. D. Thesis.

International Peer-Reviewed Conference/Proceedings

[8] B. NICOLAE, G. ANTONIU, L. BOUGÉ. *BlobSeer: How to Enable Efficient Versioning for Large Object Storage under Heavy Access Concurrency*, in "2nd International Workshop on Data Management in Peer-to-peer systems (DAMAP 2009), Saint-Petersburg, Russia", 2009, http://hal.inria.fr/inria-00382354/en/, Held in conjunction with the EDBT/ICDT 2009 Joint Conference.

[9] B. NICOLAE, G. ANTONIU, L. BOUGÉ. *Enabling High Data Throughput in Desktop Grids Through Decentralized Data and Metadata Management: The BlobSeer Approach*, in "Proc. 15th International European Conference on Parallel and Distributed Computing (Euro-Par 2009), Delft, The Netherlands", Lecture Notes in Computer Science, n^{o} 5704, TU Delft, 2009, p. 404-416, http://hal.inria.fr/inria-00410956/en/.

[10] V.-T. TRAN, G. ANTONIU, B. NICOLAE, L. BOUGÉ. *Towards A Grid File System Based On A Large-Scale BLOB Management Service*, in "CoreGRID ERCIM Working Group Workshop on Grids, P2P and Service computing, Delft, The Netherlands", 2009, http://hal.inria.fr/inria-00425232/en/, To appear. Held in conjunction with the 15th International Euro-Par Conference, Delft, The Netherlands.

Research Reports

[11] J. CAI. *BlobSeer Monitoring Service*, INRIA, 2009, http://hal.inria.fr/inria-00411369/en/, RT-0368, Technical ReportCN.

[12] A. CARPEN-AMARIE, J. CAI, L. BOUGÉ, G. ANTONIU, A. COSTAN. *Monitoring the BlobSeer distributed data-management platform using the MonALISA framework*, INRIA, 2009, http://hal.inria.fr/inria-00410216/en/, RR-7018, Research ReportCNRO.

[13] A. CARPEN-AMARIE, J. CAI, A. COSTAN, G. ANTONIU, L. BOUGÉ. *Bringing Introspection Into the BlobSeer Data-Management System Using the MonALISA Distributed Monitoring Framework*, INRIA, 2009, http://hal.inria.fr/inria-00419978/en/, RR-7043, Research ReportCNRO.

[14] L. CUDENNEC, G. ANTONIU, L. BOUGÉ. *Experimentations With CoRDAGe, A Generic Service For Co-Deploying and Re-Deploying Applications On Grids*, INRIA, 2009, http://hal.inria.fr/inria-00425555/en/, RR-7086, Research Report.

[15] B. NICOLAE, D. MOISE, G. ANTONIU, L. BOUGÉ, M. DORIER. *BlobSeer: Bringing High Throughput under Heavy Concurrency to Hadoop Map/Reduce Applications*, n^{o} RR-7140, INRIA, 2009, http://hal.inria.fr/inria-00440312/en/, A slightly revised version of this work will be published in the Proceedings of the 24th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2010), Atlanta, April 2010, Research Report.

Other Publications

[16] J. MONTES, B. NICOLAE, G. ANTONIU, A. SÁNCHEZ, MARÍA S. PÉREZ. *Using Global Behavior Modeling to Improve QoS in Large-scale Distributed Data Storage Services*, 2009, Submitted for publication.

References in notes

[17] *Chirp protocol specification*, 2009, http://www.cs.wisc.edu/condor/chirp/.

[18] *Lightweight Data Replicator*, 2009, http://www.lsc-group.phys.uwm.edu/LDR/.

[19] *Amazon Elastic Compute Cloud (EC2)*, 2009, http://aws.amazon.com/ec2/.

[20] *The Eucalyptus project*, 2009, http://open.eucalyptus.com/.

[21] *Google App Engine*, 2009, http://code.google.com/appengine/.

[22] *Google Docs*, 2009, http://www.google.com/google-d-s/tour1.html.

[23] *HadoopFS*, 2009, http://hadoop.apache.org/core/docs/current/hdfs_design.html.

[24] *Microsoft Azure*, 2009, http://www.microsoft.com/azure/.

[25] *Microsoft Office Live*, 2009, http://www.officelive.com/.

[26] *The Nimbus project*, 2009, http://workspace.globus.org/.

[27] *The XtreemOS project*, 2009, http://www.xtreemos.eu/.

[28] B. ALLCOCK, J. BESTER, J. BRESNAHAN, A. L. CHERVENAK, I. FOSTER, C. KESSELMAN, S. MEDER, V. NEFEDOVA, D. QUESNEL, S. TUECKE. *Data management and transfer in high-performance computational grid environments*, in "Parallel Comput.", vol. 28, n^{o} 5, 2002, p. 749–771, http://dx.doi.org/10.1016/S0167-8191(02)00094-7.

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

[30] G. ANTONIU, L. BOUGÉ, M. JAN. *JuxMem: An Adaptive Supportive Platform for Data Sharing on the Grid*, in "Scalable Computing: Practice and Experience", vol. 6, n^{o} 3, November 2005, p. 45–55, http://hal.inria.fr/inria-00000984.

[31] A. BASSI, M. BECK, G. FAGG, T. MOORE, J. S. PLANK, M. SWANY, R. WOLSKI. *The Internet Backplane Protocol: A Study in Resource Sharing*, in "Proc. 2nd IEEE/ACM Intl. Symp. on Cluster Computing and the Grid (CCGRID '02), Washington, DC, USA", IEEE Computer Society, 2002, 194.

[32] J. BENT, V. VENKATARAMANI, N. LEROY, A. ROY, J. STANLEY, A. ARPACI-DUSSEAU, R. ARPACI-DUSSEAU, M. LIVNY. *Flexibility, Manageability, and Performance in a Grid Storage Appliance*, in "Proc. 11th IEEE Symposium on High Performance Distributed Computing (HPDC 11)", 2002.

[33] R. BUYYA, C. S. YEO, S. VENUGOPAL. *Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities*, in "HPCC '08: Proceedings of the 2008 10th IEEE International Conference on High Performance Computing and Communications, Washington, DC, USA", IEEE Computer Society, 2008, p. 5–13, http://dx.doi.org/10.1109/HPCC.2008.172.

[34] P. H. CARNS, W. B. LIGON, R. B. ROSS, R. THAKUR. *PVFS: A Parallel File System for Linux Clusters*, in "ALS '00: Proceedings of the 4th Annual Linux Showcase and Conference, Atlanta, GA, USA", USENIX Association, 2000, p. 317–327.

[35] M. A. CASEY, F. KURTH. *Large data methods for multimedia*, in "Proc. 15th Intl. Conf. on Multimedia (Multimedia '07), New York, NY, USA", ACM, 2007, p. 6–7, http://doi.acm.org/10.1145/1291233.1291238.

[36] F. COSTA, L. SILVA, G. FEDAK, I. KELLEY. *Optimizing data distribution in desktop grid platforms*, in "Parallel Processing Letters (PPL)", vol. 18, 2008, p. 391 - 410, http://dx.doi.org/10.1142/S0129626408003466.

[37] J. DEAN, S. GHEMAWAT. *MapReduce: simplified data processing on large clusters*, in "Communications of the ACM", vol. 51, n^{o} 1, 2008, p. 107–113.

[38] A. DEVULAPALLI, D. DALESSANDRO, P. WYCKOFF, N. ALI, P. SADAYAPPAN. *Integrating parallel file systems with object-based storage devices*, in "SC '07: Proceedings of the 2007 ACM/IEEE conference on Supercomputing, New York, NY, USA", ACM, 2007, p. 1–10, http://dx.doi.org/10.1145/1362622.1362659.

[39] K. DOUGLAS, S. DOUGLAS. *PostgreSQL*, New Riders Publishing, Thousand Oaks, CA, USA, 2003.

[40] M. FACTOR, K. METH, D. NAOR, O. RODEH, J. SATRAN. *Object storage: the future building block for storage systems*, in "Local to Global Data Interoperability - Challenges and Technologies, 2005", 2005, p. 119–123, http://dx.doi.org/10.1109/LGDI.2005.1612479.

[41] S. GHEMAWAT, H. GOBIOFF, S.-T. LEUNG. *The Google file system*, in "SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles, New York, NY, USA", ACM Press, 2003, p. 29–43, http://dx.doi.org/10.1145/945445.945450.

[42] S. GRIMES. *Unstructured Data and the 80 Percent Rule*, 2008, Carabridge Bridgepoints.

[43] P. HONEYMAN, W. A. ADAMSON, S. MCKEE. *GridNFS: global storage for global collaborations*, in "Proc. IEEE Intl. Symp. Global Data Interoperability - Challenges and Technologies, Sardinia, Italy", IEEE Computer Society, June 2005, p. 111–115.

[44] M. IBRAHIM, R. ANTHONY, T. EYMANN, A. TALEB-BENDIAB, L. GRUENWALD. *Exploring Adaptation & Self-Adaptation in Autonomic Computing Systems*, in "Database and Expert Systems Applications, International Workshop on", vol. 0, 2006, p. 129-138, http://doi.ieeecomputersociety.org/10.1109/DEXA.2006.57.

[45] R. JIN, G. YANG. *Shared Memory Parallelization of Data Mining Algorithms: Techniques, Programming Interface, and Performance*, in "IEEE Trans. on Knowl. and Data Eng.", vol. 17, n^{o} 1, 2005, p. 71–89, http://dx.doi.org/10.1109/TKDE.2005.18.

[46] K. KEAHEY, T. FREEMAN. *Science Clouds: Early Experiences in Cloud Computing for Scientific Applications*, in "Cloud Computing and Its Applications 2008 (CCA-08), Chicago, IL", 2008.

[47] J. O. KEPHART, D. M. CHESS. *The Vision of Autonomic Computing*, in "Computer", vol. 36, n^{o} 1, 2003, p. 41–50, http://dx.doi.org/10.1109/MC.2003.1160055.

[48] P. Z. KUNSZT, E. LAURE, H. STOCKINGER, K. STOCKINGER. *File-based replica management*, in "Future Generation Computing Systems", vol. 21, n^{o} 1, 2005, p. 115-123.

[49] A. LENK, M. KLEMS, J. NIMIS, S. TAI, T. SANDHOLM. *What's inside the Cloud? An architectural map of the Cloud landscape*, in "Software Engineering Challenges of Cloud Computing (CLOUD '09)", 2009, p. 23 - 31, ICSE Workshop.

[50] M. MESNIER, G. R. GANGER, E. RIEDEL. *Object-based storage*, in "Communications Magazine, IEEE", vol. 41, n^{o} 8, 2003, p. 84–90, http://dx.doi.org/10.1109/MCOM.2003.1222722.

[51] C. MORIN, J. GALLARD, Y. JÉGOU, P. RITEAU. *Clouds: a new playground for the XtreemOS Grid operating system*, in "Parallel Processing Letters", vol. 19, n^{o} 3, 2009, p. 435-449, To appear.

[52] C. MORIN. *XtreemOS: a Grid Operating System Making your Computer Ready for Participating in Virtual Organizations*, in "IEEE International Symposium on Object/component/service-oriented Real-time distributed Computing (ISORC), Santorini Island, Greece", 2007.

[53] M. NICOLA, M. JARKE. *Performance Modeling of Distributed and Replicated Databases*, in "IEEE Trans. on Knowl. and Data Eng.", vol. 12, n^{o} 4, 2000, p. 645–672, http://dx.doi.org/10.1109/69.868912.

[54] C. OLSTON, B. REED, U. SRIVASTAVA, R. KUMAR, A. TOMKINS. *Pig latin: a not-so-foreign language for data processing*, in "SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, New York, NY, USA", ACM, 2008, p. 1099–1110, http://doi.acm.org/10.1145/1376616.1376726.

[55] M. PARASHAR, S. HARIRI. *Autonomic computing: An overview*, in "Unconventional Programming Paradigms", Springer Verlag, 2005, p. 247–259.

[56] A. RAGHUVEER, M. JINDAL, M. F. MOKBEL, B. DEBNATH, D. DU. *Towards efficient search on unstructured data: an intelligent-storage approach*, in "CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, New York, NY, USA", ACM, 2007, p. 951–954, http://doi.acm.org/10.1145/1321440.1321583.

[57] P. SCHWAN. *Lustre: Building a file system for 1000-node clusters*, in "Proceedings of the Linux Symposium", 2003, http://citeseer.ist.psu.edu/schwan03lustre.html.

[58] A. SEZNEC, N. SENDRIER. *HAVEGE: A User-Level Software Heuristic for Generating Empirically Strong Random Numbers*, in "ACM Transactions on Modeling and Computer Simulation", vol. 13, n^{o} 4, October 2003, p. 334–346.

[59] O. TATEBE, Y. MORITA, S. MATSUOKA, N. SODA, S. SEKIGUCHI. *Grid Datafarm Architecture for Petascale Data Intensive Computing*, in "Proc. 2nd IEEE/ACM Intl. Symp. on Cluster Computing and the Grid (Cluster 2002), Washington DC, USA", IEEE Computer Society, 2002, 102.

[60] A. THOMASIAN. *Concurrency control: methods, performance, and analysis*, in "ACM Computing Survey", vol. 30, n^{o} 1, 1998, p. 70–119, http://doi.acm.org/10.1145/274440.274443.

[61] L. M. VAQUERO, L. RODERO-MERINO, J. CACERES, M. LINDNER. *A break in the clouds: towards a cloud definition*, in "SIGCOMM Comput. Commun. Rev.", vol. 39, n^{o} 1, 2009, p. 50–55, http://doi.acm.org/10.1145/1496091.1496100.

[62] S. A. WEIL, S. A. BRANDT, E. L. MILLER, D. D. E. LONG, C. MALTZAHN. *Ceph: a scalable, high-performance distributed file system*, in "OSDI '06: Proceedings of the 7th symposium on Operating systems design and implementation, Berkeley, CA, USA", USENIX Association, 2006, p. 307–320, http://portal.acm.org/citation.cfm?id=1298455.1298485.

[63] B. S. WHITE, M. WALKER, M. HUMPHREY, A. S. GRIMSHAW. *LegionFS: a secure and scalable file system supporting cross-domain high-performance applications*, in "Proc. 2001 ACM/IEEE Conf. on Supercomputing (SC '01), New York, NY, USA", ACM Press, 2001, p. 59–59.