INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Team Regal

Resource management in large scale distributed systems

Paris - Rocquencourt

THEME COM

Table of contents

1. Team

Regal is a common project with CNRS and Unviersit of Paris 6 through the "Laboratoire d'Informatique de Paris 6", LIP6 *(UMR 7606).*

Head of project-team
Pierre Sens [Professor Univeristé Paris 6, HdR]

Vice-head ofteam
Marc Shapiro [Research Director (DR) INRIA, HdR]

Administrative assistant
Pascale Tabard [Secretary Inria]

Research scientist Inria
Mesaac Makpangou [Research associate (CR) Inria, HdR]

Research scientist (external)
Luciana Arantes [Associate professor Université Paris 6]
Maria Gradinariu [Associate professor Université Paris 6]
Bertil Folliot [Professor Université Paris 6, HdR]
Olivier Marin [Associate professor Université Paris 6]
Sebastien Monnet [Associate professor Université Paris 6 since September 2007]
Gaël Thomas [Associate professor Université Paris 6]

Engineer
Jean-Michel Busca [Research Engineer]
Véronique Martin [Associate Engineer]

Visiting scientist
Denis Conan [3 months Sabbatical at Regal - Associate professor INT, Evry, France, for January till March 31]

Ph. D. student
Lamia Benmouffok [Microsoft research grant - Université Paris 6]
Mathieu Bouillaguet [Université Paris 6]
Ikram Chabbouh [Université de Tunis]
Charles Clement [Université Paris 6]
Nicolas Geoffray [Université Paris 6]
Cyril Martin [Université Paris 6]
Corentin Mehat [Université Paris 6]
Julien Sopena [Université Paris 6]
Pierre Sutra [Université Paris 6]
Mathieu Valéro [Université Paris 6]

2. Overall Objectives

2.1. Overall Objectives

The main focus of the Regal team is research on large-scale distributed computing systems, and addresses the challenges of automated adminstration of highly dynamic networks, of fault tolerance, of information sharing in collaborative groups, of dynamic content distribution, and of operating system adaptation. Regal is a joint research team between LIP6 and INRIA-Paris-Rocquencourt.

2.2. Highlights

- Participation to the new PLAY ALL project
- Majors publications in 2007 : Distributed Computing Journal [14], IEEE TNSM Journal [16], 2 publications in ICDCS conference [25], [19], ICPP conference [30], CollaborateCom conference [27].
- Sebastien Monnet (Assistant professor) has joined the team
- 2 PhD defenses : Ikram Chabbouh (dec. 2007), Jean-Michel Busca (dec. 2007)

3. Scientific Foundations

3.1. Scientific Foundations

Keywords: *Grid computing*, *Peer-to-peer*, *consistency*, *distributed system*, *dynamic adaptation*, *fault tolerance*, *large scale environments*, *replication*.

Scaling to large configurations is one of the major challenges addressed by the distributed system community lately. The basic idea is how to efficiently and transparently use and manage resources of millions of hosts spread over a large network. The problem is complex compared to classical distributed systems where the number of hosts is low (less than a thousand) and the inter-host links are fast and relatively reliable. In such "classical" distributed architectures, it is possible and reasonable to build a single image of the system so as to "easily" control resource allocation.

In large configurations, there is no possibility to establish a global view of the system. The underlying operating system has to make decisions (on resource allocation, scheduling ...) based only on partial and possibly wrongs view of the resources usage.

Scaling introduces the following problems:

- Failure: as the number of hosts increases, the probability of a host failure converges to one. For instance if we consider a classical host MTBF (Mean Time Between Failure) equals to 13 days, in a middle scale system composed of only 10000 hosts, a failure will occur every 4 minutes. Compared to classical distributed systems, failures are more common and have to be efficiently processed.
- Asynchronous networks: clearly on the Internet network transmission delays are very dynamic and unbounded. The asynchronous nature of the Internet makes it extremely problematic to control the consistency of the system. This issue has its roots in the Fischer-Lynch-Patterson impossibility result that shows that a basic feature such as consensus cannot be deterministically solved in an asynchronous system subject to only one crash failure. The fundamental difficulty is that the system has to compose on possibly wrong views of the system where hosts suspected faulty are correct, or where really faulty host are seen in a correct state.
- Failure models: classical distributed systems usually consider crash or omission failures. In the context of large-scale networks, such as peer-to-peer overlays, such an assumption is not reasonable since hosts can not only be affected by failures but can be attacked and consequently become malicious. Clearly the failure model has to be extended to deal with a possibly Byzantine behavior of hosts.

Two architectures in relation with the scaling problem have emerged during the last years:

Grid computing: Grid computing offers a model for solving massive computational problems using large numbers of computers arranged as clusters interconnected by a telecommunications infrastructure as internet, renater or VTHD.

If the number of involved hosts can be high (several thousands), the global environment is relatively controlled and users of such systems are usually considered safe and only submitted to host crash failures (typically, Byzantine failures are not considered).

Peer-to-peer overlay network: Generally, a peer-to-peer (or P2P) computer network is any network that does not rely on dedicated servers for communication but, instead, mostly uses direct connections between clients (peers). A pure peer-to-peer network does not have the notion of clients or servers, but only equal peer nodes that simultaneously function as both "clients" and "servers" with respect to the other nodes on the network.

This model of network arrangement differs from the client-server model where communication is usually relayed by the server. In a peer-to-peer network, any node is able to initiate or complete any supported transaction with any other node. Peer nodes may differ in local configuration, processing speed, network bandwidth, and storage capacity.

Different peer-to-peer networks have varying P2P overlays. In such systems, no assumption can be made on the behavior of the host and Byzantine behavior has to be considered.

Regal is interested in how to adapt distributed middleware to these large scale configurations. We target Grid and Peer-to-peer configurations. This objective is ambitious and covers a large spectrum. To reduce its spectrum, Regal focuses on fault tolerance, replication management, and dynamic adaptation.

Basically, Regal proposes the use of *reactive replication* to tolerate faults and to reduce the access time to get data, while adapting dynamically to the environmental constraints and the evolution of application behavior.

We concentrate on the following research themes:

Data management: the goal is to be able to deploy and locate effectively data while maintening the required level of consistency between data replicas.

System monitoring and failure detection: we envisage a service providing the follow-up of distributed information. Here, the first difficulty is the management of a potentially enormous flow of information which leads to the design of dynamic filtering techniques. The second difficulty is the asynchronous aspect of the underlying network which introduces a strong uncertainty on the collected information.

Adaptive replication: we design parameterizable techniques of replication aiming to tolerate the faults and to reduce information access times. We focus on the runtime adaptation of the replication scheme by (1) automatically adjusting the internal parameters of the strategies and (2) by choosing the replication protocol more adapted to the current context.

The dynamic adaptation of application execution support: the adaptation is declined here to the level of the execution support (in either of the high level strategies). We thus study the problem of dynamic configuration at runtime of the low support layers.

4. Application Domains

4.1. Application Domains

Keywords: *Internet services*, *data storage*, *data-sharing*, *multi-agent systems*.

As we already mentioned, we focus on two kinds of large scale environments: computational grids and peer-to-peer (P2P) systems. Although both environments have the same final objective of sharing large sets of resources, they initially emerged from different communities with different context assumptions and hence they have been designed differently. Grids provide support for a large number of services needed by scientific communities. They usually target thousands of hosts and hundreds of users. Peer-to-peer environments address millions of hosts with hundreds of thousands of simultaneous users but they offer limited and specialized functionalities (file sharing, parallel computation).

In peer-to-peer configurations we focus on the following applications:

- Internet services such as web caches or content distribution network (CDN) which aim at reducing the access time to data shared by many users,
- Data storage of mutable data. Data storage is a classical peer-to-peer application where users can share documents (audio and video) across the Internet. A challenge for the next generation of data sharing systems is to provide update management in order to develop large cooperative applications.

In Grid configurations we address resource management for two kinds of applications:

- Multi-agent applications which model complex cooperative behaviors.
- Application Service Provider (ASP) environments in cooperation with the DIET project of the GRAAL team.

Our third application domain is based on data sharing. Whereas most work on P2P applications focuses on write-once single-writer multiple-reader applications, we consider the (more demanding) applications that share mutable data in large-scale distributed settings. Some examples are co-operative engineering, collaborative authoring, or entreprise information libraries: for instance co-operative code development tools or decentralised wikis. Such applications involve users working from different locations and at different times, and for long durations. In such settings, each user *optimistically* modifies his private copy, called a replica, of a shared datum. As replicas may diverge, this poses the problem of reconciliation. Our research takes into account a number of issues not addressed by previous work, for instance respecting application semantics, high-level operations, dependence, atomicity and conflict, long session times, etc.

5. Software

5.1. Pastis: A peer-to-peer file system

Participants: Pierre Sens [correspondent], Jean-Michel Busca.

Pastis is a distributed multi-writer file system. It aims at making use of the aggregate storage capacity of hundreds of thousands of PCs connected to the Internet by means of a completely decentralized peer-to-peer (P2P) network. Replication allows persistent storage in spite of a highly transient node population, while cryptographic techniques ensure the authenticity and integrity of file system data.

Routing and data storage in Pastis are handled by the Pastry routing protocol and the PAST distributed hash table (DHT). The good locality properties of Pastry/PAST allow Pastis to minimize network access latencies, thus achieving a good level of performance when using a relaxed consistency model. Moreover, Pastis does not employ heavy protocols such as BFT (Byzantine Fault Tolerance), like other P2P multi-writer file systems do. In Pastis, for a file system update to be valid, the user must provide a certificate signed by the file owner which proves that he has write access to that file.

5.2. LS3: Large Scale Simulator

Participants: Pierre Sens [correspondent], Jean-Michel Busca.

LS3 is a discrete event simulator originally developped for Pastis, a peer-to-peer file system based on Pastry. LS3 allows to build a network of tenths of thousands nodes on a single computer, and simulate its execution by taking into account message transmission delays. LS3 transparently simulates communication layers between nodes, and executes the same application code (including Pastry, Past and higher layers) as in a real execution of the system.

LS3's modular design consists of three independent layers, allowing the simulator to be reused in areas other than Pastis and Pastry:

- At the kernel level, the system being simulated is described in a generic way in terms of entities triggered by events. Each entity has a current state and a current virtual time, and can be programmed either in synchronous mode (blocking wait of the next event) or in asynchronous mode (activation of an event handler). A multi-threaded event engine delivers events in chronological order to each entity by applying a conservative scheduling policy, based on the analysis of event dependencies.
- At the network level, the system being simulated is modeled in terms of nodes sending and receiving messages, and connected through a network. The transmission delay of a message is derived from the distance between the sending and the receiving nodes in the network, according to the selected topology. Three topologies can be used: local network (all nodes belong to the same local network), two-level hierarchy (nodes are grouped into LANs connected through WANs) and sphere (nodes are located on a sphere). It is possible to set the jitter rate of transmission delays, as well as the rate of message loss in the network.
- The stubs Pastry level interfaces Pastry with LS3: it defines a specialization of Pastry nodes that allows them to interface with standard LS3 nodes. Several parameters and policies that drive the behaviour and the structure of a Pastry network can be set at this level, including: the distribution of node ids, the selection of boostrap nodes, the periodicity of routing tables checks and the rate of node churn. It is also possible to simulate the ping messages that nodes send to supervise each other, and set failure detection thresholds.

Some figures: LS3 can simulate a network of 20 000 Pastry nodes with no application within 512 Mb of RAM, and it takes approximately 12 minutes on a single processor Pentium M 1,7 GHz to build such network. When simulating the Pastis application, event processing speed is about 500 evt/s. The speedup factor depends on the simulated load: as an example, speedup ranges from 20 for a single user to 0,05 for 400 simultaneous users.

5.3. Telex

Participants: Marc Shapiro [correspondent], Lamia Benmouffok, Jean-Michel Busca, Pierre Sutra.

Regal is developping a middleware for optimistic replication and reconciliation, named Telex. Telex is based on a high-level formalism for replication (the Action-Constraint Framework). It is designed to easily express a large spectrum of different replication protocols, and to support them efficiently.

Within the Respire and Grid4All projects, Telex will be used for a number of collaborative applications such as shared editors and shared decision making. Telex shared documents are based on a persistent and replicated data structure called the multilog, which consists of an ever-growing graph of action nodes (operations) linked by constraint edges (semantic relations). Multilogs simplify consistency maintenance and enable novel modes of collaboration.

The main issues for multilogs are the following: to design a multilog format that avoids contention on both reads and writes; to design an interface that is both sufficiently expressive and makes efficient use of bandwidth; to record sufficient information without wasting space; to study the trade-off between checkpointing and logging; and to enable garbage collection.

A central design decision was to store a multilog as a set of append-only logs and to create a separate file for each user session or once the previous chunk has reached a maximum size. I/O and storage maintain the boundaries between atomic sets of actions and constraints. In order to optimise network and storage bandwidth, objects use custom persistence.

6. New Results

6.1. Introduction

In 2007, we focused our research on the following areas:

- distributed algorithms for Grid configuration,
- Peer-to-peer storage
- dynamic adaptation of virtual machines,
- services management in large scale environments,
- Formal and practical study of optimistic replication, incorporating application semantics.
- Decentralised commitment protocols for semantic optimistic replication.
- dynamic replication in distributed multi-agent systems.
- self* systems

6.2. Distributed algorithms

Participants: Luciana Arantes [correspondent], Mathieu Bouillaguet, Pierre Sens, Julien Sopena.

Our current research focuses on adapting distributed algorithms to large-scale and heterogeneous environments. We target particularly three basic blocks: mutual exclusion algorithms, resource allocation and failure detection.

- In mutual exclusion algorithm, we have been interested in scalability, fault-tolerance and latency tolerance aspects of distributed mutual exclusion algorithms.

  The majority of current distributed mutual exclusion algorithms are not suited for distributed applications on top of large-scale or heterogeneous systems such as Grid or peer-to-peer systems since they do not consider the latency heterogeneity of the platform or scalable fault tolerance mechanisms. In [30] we propose a new composition approach to mutual exclusion algorithms for applications spread over a grid which is composed of a federation of clusters. Taking into account the heterogeneity of communication latency, our hierarchical architecture combines intra and inter cluster algorithms. We focus on token-based algorithms and study different compositions of algorithms. Performance evaluation tests have been conducted on a national grid testbed whose results show that our approach is scalable and that the choice of the most suitable inter cluster algorithm depends on the behavior of the application.

- Failure detectors are well-known as a basic building block of fault-tolerant distributed systems. A unreliable failure detector can be informally considered as an oracle per process. Such an oracle provides the set of processes that it currently suspects of having crashed. Many applications use a failure detection service in order to solve the consensus problem. Typically, failure detectors use messages sent periodically between hosts. Most of implementations are based on all-to-all communication where each process periodically sends an I-am-alive message to all processes in order to inform them that it is still alive, and thus resulting in a quadratic number of message to be periodically send.

  We previously proposed a scalable implementation of failure detector (called GFD for *Grid Failure Detector*) [5] taking into account the network topology. We consider now the problem of failure detection in dynamic network such as MANET or Peer-to-peer overlay [36]. Unreliable failure detectors (FD) are classical mechanisms providing information about process failures. They allow to solve consensus in asynchronous network. However, most of implementations consider a set of known processes fully connected by reliable links. Such an assumption is not applicable in dynamic environments. Furthermore, the majority of current failure detector implementation are timer-based ones while in dynamic networks there is not an upper bound for communication delays. We propose an asynchronous implementation of a failure detector for dynamic environments. Our implementation is an query-response algorithm which does not rely on timers to detect failures. We assume that neither the identity nor the number of nodes are initially known. We also proof that our algorithm can implement failure detectors of class $\diamondsuit S$ when some behavioral properties are satisfied by the underlying system. Simulation results have validated our approach.

6.3. Peer-to-peer systems

Participants: Pierre Sens [correspondent], Jean-Michel Busca, Sebastien Monnet.

6.3.1. Peer-to-peer storage

Since 2003, we develop Pastis [6] is a new completely decentralized multi-user read-write peer-to-peer file system. Pastis is based on the FreePastry Distributed Hash Table (DHT) of the Rice University. DHTs provide a means to build a completely decentralized, large-scale persistent storage service from the individual storage capacities contributed by each node of the peer-to-peer overlay

However, persistence can only be achieved if nodes are highly available, that is, if they stay most of the time connected to the overlay. Churn (i.e., nodes connecting and disconnecting from the overlay) in peer-to-peer networks is mainly due to the fact that users have total control on theirs computers, and thus may not see any benefit in keeping its peer-to-peer client running all the time. In 2007, we study the effects of churn on Pastis [35], a DHT-based peer-to-peer file system. We evaluate the behavior of Pastis under churn, and investigate whether it can keep up with changes in the peer-to-peer overlay. We used a modified version of the PAST DHT to provide better support for mutable data and to improve tolerance to churn. Our replica regeneration protocol distinguishes between mutable blocks and immutable blocks to minimize the probability of data loss. Read-write quorums provide a good compromise to ensure replica consistency under the presence of node failures. Our experiments use Modelnet to emulate wide-area latencies and the asymmetric band- width of ADSL client links. The results show that Pastis preserves data consistency even at high levels of churn.

Recently, we consider the problem of data durability in wide-area distributed storage systems, such as Distributed Hash Tables (DHTs). Given the limited bandwidth between replicas, these systems suffer from long repair times after a hard disk crash, making them vulnerable to data loss when several replicas fail within a short period of time. Recent work has suggested that the probability of data survival can be predicted by modeling the number of live replicas using a Markov chain. However, the prediction accuracy relies on a realistic estimation of the model parameters, that is, the chain transition rates from one state to another. In [29], [28], we focus on obtaining a good approximation of the Markov chain parameters used to predict data durability. We present a new set of expressions for these parameters based on a theoretical analysis and on empirical data. We argue that these parameters produce a conservative estimation of the probability of object survival. In other words, our expression allows the system designer to choose the appropriate replication factor to guarantee a minimum level of data durability. To the best of our knowledge, this is the first work that focuses on this parameter estimation problem.

We also study peer-to-peer meta-data management for knowledge discovery applications in grids. Computational Grids are powerful platforms gathering computational power and storage space from thousands of geographically distributed resources. The applications running on such platforms need to efficiently and reliably access the various and heterogeneous distributed resources they offer. This can be achieved by using meta-data information describing all available resources. It is therefore crucial to provide efficient meta-data management architectures and frameworks. We have designed and implemented a Grid meta-data management service. We focused on a particular use case: the Knowledge Grid architecture (University of Calabria) which provides high-level Grid services for distributed knowledge discovery applications. Taking advantage of an existing Grid data-sharing service, namely JuxMem (IRISA/INRIA, Rennes), the proposed solution lies at the border between peer-to-peer systems and Web services. It has been experimented on the Grid'5000 french testbed.

6.3.2. Peer-to-peer overlay

A malleable peer-to-peer overlay to take into account applications needs. Peer-to-peer overlays allow distributed applications to work in a wide-area, scalable, and fault-tolerant manner. However, most structured and unstructured overlays present in literature today are inflexible from the application viewpoint. In other words, the application has no control over the structure of the overlay itself. This paper proposes the concept of an application-malleable overlay, and the design of the first malleable overlay which we call MOve. In MOve, the communication characteristics of the distributed application using the overlay can influence the

overlay's structure itself, with the twin goals of (1) optimizing the application performance by adapting the overlay, while also (2) retaining the scale and fault-tolerance of the overlay approach. The influence could either be explicitly specified by the application or implicitly gleaned by our algorithms. Besides neighbor list membership management, MOve also contains algorithms for resource discovery, update propagation, and churn-resistance. The emergent behavior of the implicit mechanisms used in MOve manifest in the following way: when application communication is low, most overlay links keep their default configuration; however, as application communication characteristics become more evident, the overlay gracefully adapts itself to the application.

6.4. Self* properties of dynamic systems

Participant: Maria Gradinariu [correspondent].

The main challenges of our research activity over 2007 year were to develop self* (self-stabilizing, self-organizing and self-healing) algorithms for static and dynamic networks (P2P, sensor and robot networks). Each of these systems has its own specificity in terms of network characteristics. Therefore algorithms for these systems should be adapted to the network specificity. For example in P2P systems nodes can communicate with the nodes in the acquittance neighborhood while in sensor and robot networks the communication is restricted to the communication range limited by the physical communication power of each node. We addressed fundamental problems such as communication primitives and infrastructures in P2P and sensor networks, leader election and pattern formation in robot networks and conflict managers and leader election for static networks. All the proposed algorithms are self* and cope with the network dynamicity in the case of P2P, sensor or robot networks.

6.4.1. Services for dynamic networks

Self* communication primitives and infrastructures for P2P and Sensor Networks. In [20], [19], we propose and prove correct a distributed stabilizing implementation of an overlay, called DR-tree, optimized for efficient selective dissemination of information. DR-tree copes with nodes dynamicity (frequent joins and leaves) and memory and counter program corruptions, that is, the processes can connect/disconnect at any time, and their memories and programs can be corrupted. The maintenance of the structure is local and requires no additional memory to guarantee its stabilization. The structure is balanced and is of height O(logm(N)), which makes it suitable for performing efficient data storage or search. We extend our overlay in order to support complex content-based filtering in publish/subscribe systems. Publish/ subscribe systems provide useful platforms for delivering data (events) from publishers to subscribers in a decoupled fashion in distributed networks. Developing efficient publish/subscribe schemes in dynamic distributed systems is still an open problem for complex subscriptions (spanning multi-dimensional intervals). Embedding a publish/ subscribe system in a DR-trees is a new and viable solution. The DR-tree overlay also guarantees subscription and publication times logarithmic in the size of the network while keeping its space requirement low (comparable to its DHT-based counterparts). Nonetheless, the DRtree overlay helps in eliminating the false negatives and drastically reduces the false positives in the embedded publish/ subscribe system.

Sensor networks are mainly used to gather strategic information in various monitored areas. Sensors may be deployed in zones where their internal memory, or the sensors themselves, can be corrupted. Since deployed sensors cannot be easily replaced, network persistence and robustness are the two main issues that have to be addressed while efficiently deploying large scale sensor networks. The goal of forming a Minimum Connected Cover of a query region in sensor networks is to select a subset of nodes that entirely covers a particular monitored area, which is strongly connected, and which does not contain a subset with the same properties. In this paper, we consider the general case, wherein every sensor has a different sensing and communication radius. In [24] we propose two novel and robust solutions to the minimum connected cover problem that can cope with both transient faults (corruptions of the internal memory of sensors) and sensor crash/join. Also, our proposal includes extended versions which use multi-hop information. Our algorithms use small atomicity (i.e., each sensor reads variables of only one of its neighbors at a time). Our solutions are self* (self-configuration, self-stabilization, and self-healing). Via simulations, we conclude that our solutions provide

better performance, in terms of coverage, than pre-existing self-stabilizing solutions. Moreover, we observe that multi-hop solutions produce a better approximation to an optimal cover set.

Leader election, Naming and Patern formation in Robot Networks. In [21] we are interested in robot networks where the robots motion is restricted to particular types of graphs and robots do not have distinct identifiers. In these networks we address the problems of naming (robots choose an unique name in a distributed way) and leader election (one of the robots is the leader and the others are the followers).

In [23] we address leader election and pater formation problems in robot netwoks where robots do not have motion or visibility restrictions. Flocking is the ability of a group of robots to follow a leader or head whenever it moves in a plane (two dimensional Cartesian space). In this paper we propose and prove correct an architecture for a self-organizing and stabilizing flocking system. Contrary to the existing work on this topic our flocking architecture does not rely on the existence of a specific leader a priori known to every robot in the network. In our approach robots are uniform, start in an arbitrary configuration and the head of the group is elected via algorithmic tools. Our contribution is threefold. First, we propose novel probabilistic solutions for leader election in asynchronous settings under bounded schedulers. Additionally, we prove the impossibility of deterministic leader election when robots have no common coordinates and start in an arbitrary configuration. Secondly, we propose a collision free deterministic algorithm for circle formation designed for asynchronous networks. Thirdly, we propose a deterministic flocking algorithm totally independent of the existence of an a priori known leader. The proposed algorithm also works in asynchronous networks.

6.4.2. Services for Static networks

Self-stabililizing leader election and conflict managers for static networks. In [25] we specify the conflict manager abstraction. Informally, a conflict manager guarantees that any two nodes that are in conflict cannot enter their critical section simultaneously (safety), and that at least one node is able to execute its critical section (progress). The conflict manager problem is strictly weaker than the classical local mutual exclusion problem, where any node that requests to enter its critical section eventually does so (fairness). We argue that conflict managers are a useful mechanism to transform a large class of self-stabilizing algorithms that operate in an essentially sequential model, into selfstabilizing algorithm that operate in a completely asynchronous distributed model. We provide two implementations (one deterministic and one probabilistic) of our abstraction, and provide a composition mechanism to obtain a generic transformer. Our transformers have low overhead: the deterministic transformer requires one memory bit, and guarantees time overhead in order of the network degree, the probabilistic transformer does not require extra memory. While the probabilistic algorithm performs in anonymous networks, it only provides probabilistic stabilization guarantees. In contrast, the deterministic transformer requires initial symmetry breaking but preserves the original algorithm guarantees.

In [14] we present a randomized self-stabilizing leader election protocol and a randomized self-stabilizing token circulation protocol under an arbitrary scheduler on anonymous and unidirectional rings of any size. These protocols are space optimal. We also give a formal and complete proof of these protocols. To this end, we develop a complete model for probabilistic self-stabilizing distributed systems which clearly separates the non deterministic behavior of the scheduler from the randomized behavior of the protocol. This framework includes all the necessary tools for proving the self- stabilization of a randomized distributed system: definition of a probabilistic space and definition of the self-stabilization of a randomized protocol. We also propose a new technique of scheduler management through a self-stabilizing protocol composition (cross-over composition). Roughly speaking, we force all computations to have a fairness property under any scheduler, even under an unfair one.

6.4.2.1. Fault tolerant distributed agreement

In [13] we present a family of agreement problems called Managed Agreement, which is parameterized by the number of aristocrat nodes in the system; NBAC is a special case of this family when all nodes are aristocrats while Consensus is a special case of this family when there are no aristocrats. The paper also presents a parameterized family of failure detectors, the weakest failure detector class that enables solving Managed Agreement with a set A of aristocrats in an asynchronous environment.

6.5. Virtual virtual machine (VVM)

Participants: Bertil Folliot [correspondent], Charles Clement, Nicolas Goeffray, Assia Hachichi, Cyril Martin, Gaël Thomas.

The VVM group works on flexible operating systems based on a flexible dynamic compiler (JIT), free of any predefined abstractions. In 2007, VVM group focused on the following themes:

VMKit: With the proliferation of virtual machines or scripted applications, operating systems execute many virtual execution environments concurrently. This presents two main problems: (i) resource consumption and (ii) duplication of language and/or system functionalities. With VMKit, we try to extract a set of common components for all these environments that can constitute a core execution engine for all virtual execution environments. One can add new system requirements at this level: they have to be developed once for all virtual execution environments. Four complete, developed from scratch, virtual execution environments (a JVM, a CLR, a lisp-like virtual machine and an XML execution engine) already execute upon VMKit. VMKit is a flexible system that can be enrich with new system components on the fly. We developed partially a process migration system for VMKit that relies on type informations collected during dynamic compilation to allows migration on heterogeneous hardware and systems. This component is shared by our four virtual machines. First works around new system requirements are also in progress: isolation and transaction.

EndoKernel: Today, virtual machines and operating systems are two disjoints worlds: a virtual machine is executed on top of an operating system and is independent of it. This separation has two consequences: (i) first, virtual machines have only a partial view of the machine and can not manage resource at a fine-grain, (ii) second, operating systems does not take advantage of possibilities offered by dynamic compiler, such as type-safe proved compilation or independence from the hardware. EndoKernel is an experimental kernel of operating system based on VMKit. EndoKernel executes on bare hardware or as a module inside a Linux. EndoKernel relies on VMKit and allows the execution of our four virtual machines in the kernel space. In 2007, we have developed a first version of EndoKernel. This first prototype will be used to show how enforcing isolation between the kernel and its module by using our type-safe compiler, without degrading performances, and to experiment new garbage collectors algorithms based on the Memory Mangment Unit of the processor.

6.6. Replication of web applications

Participants: Mesaac Makpangou [correspondent], Ikram Chabbouh.

Web acceleration is a crutial issue for web applications, especially in e-commerce. Content caching is a well-established solution that has been used with succes for static content. Web applications serve dynamic content that are generated by programs based on parameters passed by requesters, environment variables and data stored in databases in the backend. To reuse a previously generated response to serve a new request, one must be sure that this is identical to one which would have been returned by the web application if it were invoked. Caching dynamic content raises three difficult issues: the definition of cached content unique identifiers and their freshness, and the handling of fragmentation nature of dynamic content.

Cached Content Unique Identifiers: One can reuse a cached response if both the new and the old requests lead to run the same program with the same parameters. The content identifier must refer the program, the values of parameters and variables used for its generation. The problem arise because certain parameters passed by requesters are not used at all or are not used all the times to generate the content; hence, different sets of parameters may lead to the generation of the same content. Enforcing name desambiguity of cached responses (i.e. guaranteeing that identical responses will not be attached different names and conversely that different contents are not shared the same name) is an essential feature to cache efficiently dynamically generated contents. Traditional web accelerators lack such a feature.

Cached Content Freshness: Over the time, changes will occur on the databases. As a consequence, certain responses that were previously generated and cached are now deemed obsolete. Hence, the capability to monitor changes that occur in databases in the backend, the determination of which cached

responses are impacted by these changes, and the provision of a mechanism to invalidate obselate cached responses are three essential issues to address.

Fragmentation of dynamic content: Almost all dynamic content are made of independent fragments with different lifecycles. It has been pointed by may researchers that one could improve a lot the reusability of content if fragments are made first-level fectchable contents. The challenge here is to device a tool capable to fragment efficiently an application or a dynamic content into pertinent fragments, without augmenting the burden on application programmers and without japordizing the security of the content provider.

This year we finalize FRACS, a web acceleration solution for e-commerce web applications. FRACS address the issues mentioned above. Precisely, FRACS is made of two components: FRACS Adaptator and FRACS CDN. FRACS Adaptor is a tool that automatically transforms web applications into replicable fragmented applications [12]. FRACS CDN permits to replicate fragments together with the tables of databases that they are accessing at the edges of network nearby clients.

6.7. Optimistic replication, reconciliation and commitment

Participants: Marc Shapiro [correspondent], Lamia Benmouffok, Pierre Sutra.

In distributed systems, information is replicated. Data replication enables cooperative work, improves access latency to data shared through the network, and improves availability in the presence of failures. When the information is updated, maintaining consistency between replicas is a major challenge.

Previous studies of data replication considered different areas separately, often ignoring the requirements of other areas. For instance, OS researchers often assume updates are independent; CSCW researchers ignore conflicts; algorithms research mostly ignores semantics; peer-to-peer systems often ignore mutable data and hence consistency; none of the above have addressed partial replication.

We study optimistic replication for multi-user collaborative applications such as co-operative engineering (e.g., co-operative code development), collaborative authoring (e.g., a decentralised wikipedia), or entreprise information libraries. We propose a general-purpose approach, subsuming the previous work in different areas. It takes addresses respecting application semantics, high-level operations, dependence, atomicity and conflict, long session times, etc.

Earlier, we proposed the Action-Constraint Framework (ACF) formalism to reify application semantics and to reason about reconciliation. ACF represents user intents and application semantics as constraints i.e., invariants between actions (operations) that each local scheduler must maintain. Consistency is defined abstractly by two properties, mergeability (safety) and eventual decision (liveness). Our IceCube reconciliation engine used ACF to select promising reconciliation combinations and propose them to the user. However, IceCube was not compatible with a peer-to-peer approach, because it assumed that there is a primary reconciliation site. Decentralised reconciliation algorithms exist, but they ignore semantic relations.

Therefore, we now take a different approach. Any number of "proposer" peers make reconciliation proposals (computed using IceCube, some other algorithm, or even manually). Any number of "acceptor" peers receive the proposals, which can be broken into well-formed granules called candidates. Acceptors exchange votes regarding the candidates. Candidates that have certain common characteristics compete in an election. The winner of an election is the candidate with the highest number of votes (not necessarily a majority). Every acceptor eventually unambiguously learns the outcome of each election. The protocol is completely decentralised and asynchronous. In the presence of crashes, it remains safe, and is live as long as a sufficient number of votes remain available. This work was published at Coopis [31]..

We are also working on building Telex, a general-purpose peer-to-peer middleware, based on ACF and IceCube and a successor to our previous platform (Joyce). Telex will serve both as an experimental testbed for our research on optimistic replication, and as an implementation platform for co-operative applications. Telex is designed as a general-purpose consistency layer on top of a distributed file system such as Pastis. The main abstraction of Telex is the multilog, which encapsulates both a multicast communication channel

between peers that are co-operating over some document, and persistent storage of actions, constraints, and versions. The multilog model offers unprecedented flexibility, allowing each user to tailor a local view of shared information.

We have also started addressing issues such as partial replication and improved fault tolerance. The work described above takes place in the context of several joint projects: Grid4All, Respire and Recall.

6.8. Optimistic approaches on collaborative editing

Participant: Marc Shapiro [correspondent].

6.8.1. A comparison of optimistic approaches to collaborative editing of Wiki pages

In recent years, the Web has seen an explosive growth of massive collaboration tools, such as wiki and weblog systems. By the billions, users may share knowledge and collectively advance innovation, in various fields of science and art. Existing tools, such as the MediaWiki system for wikis, are popular in part because they do not require any specific skills. However, they are based on a centralised architecture and hence do not scale well. Moreover, they provide limited functionality for collaborative authoring of shared documents.

At the same time, peer-to-peer (P2P) techniques have grown equally explosively. They enable massive sharing of audio, video, data or any other digital content, without the need for a central server, and its attendant administration and hardware costs. P2P systems provide availability and scalability by replicating data, and by balancing workload among peers. However, current P2P networks are designed to distribute only immutable documents.

A natural research direction is to use P2P techniques to distribute collaborative documents. This raises the issue of supporting collaborative edits, and of maintaining consistency, over a massive population of users, shared documents, and sites. Within the Recall collaboration, we studied a number of alternative P2P, decentralised approaches, applied to collaborative wiki editing, contrasted with current centralised systems. Specifically, we detail P2P broadcast techniques, and we compare the existing centralised approach (MediaWiki) with several distributed, peer-to-peer approaches, namely: an operational transformation approach (MOT2), a commutativity-oriented approach (WOOTO) and a serialisation and conflict resolution approach (ACF). We evaluate each approach according to a number of specified qualitative and quantitative metrics. This work is published in CollaborateCom [27].

6.8.2. Designing a commutative replicated data type for collaborative editing: the Treedoc

To share information, users located at several sites may concurrently update a common object, e.g., a text document. Each user operates on a separate *replica* (i.e., local copy) of the document. A well-studied example is co-operatively editing a shared text.

As users make modifications, replicas diverge from one another. Operations initiated on some site propagate to other sites and are *replayed* there. Eventually every site executes every action. Even so, if sites execute them in different orders, their replicas might still not converge. Various solutions are available in the literature; for instance, serialising the actions or operational transformation. Such designs are usually complex and non-scalable; thus, despite an extensive literature, there is still no satisfactory solution to the shared text editing problem.

We suggest a different approach: design replicated data types such that operations commute with one another. Let us call such a type a *commutative replicated data type* or CRDT. CRDT replicas provably converge. Furthermore, CRDTs support a restricted form of transactions “for free.” However, designing a non-trivial CRDT is difficult.

Although the advantages of commutativity are well known, the problem of designing data types for commutativity has been neglected. We studied the design of a non-trivial CRDT for concurrent editing, called *treedoc*. Since it is a CRDT, convergence is guaranteed. It supports block operations. Space overhead is kept to a minimum: there is no to little internal meta-data; deleted information can be forgotten; and identifiers are kept short. Common edit operations respond locally and suffer no network latency. Treedoc is fault-tolerant and supports disconnected operation.

Ordinary editing operations are non-destructive and identification does not change with concurrent edits, but our implementation is very different from WOOT. The basic treedoc structure is a binary tree of *terminals* with extensions for concurrent insert. A terminal is a single character, or some other object whose internal structure is not known or not of interest, for instance a JPEG graphic. The identifier of a terminal is the path to its node. For efficiency, structural operations switch between a flat buffer and a tree. These operations are potentially non-commutative; to avoid this problem, structure changes rely either on common knowledge or on consensus. To avoid the latency associated with consensus, it occurs in the background (not in the critical path of editing operations) and aborts if it conflicts with an edit.

In summary, our contributions are the following:

- A design principle: concurrent operations should commute. We prove that *any* Commutative Replicated Data Type (CRDT) converge, under some simple and standard assumptions.
- We identify two alternative approaches to commutativity: operation coalescing vs. precedence. Coalescing is better, but is not always possible.
- The design of treedoc, a non-trivial, space-efficient, responsive, coalescing CRDT for distributed editing.
- We identify some previously-published techniques for coalescing, such as non-destructive update and invariant identity. We propose a novel implementation of these techniques.
- We propose a novel technique for coalescing: where a consensus is necessary, it is restricted to non-essential operations, occurs in the background, and aborts if it conflicts with an essential operation.
- We show that a CRDT readily supports restricted transactions at a very low implementation cost.

This work is published as a technical report [37] and has been submitted for publication.

6.9. Dynamic replication in multi-agent systems

Participants: Olivier Marin [correspondent], Pierre Sens.

Distributed agent systems stand out as a powerful tool for designing scalable software. The general outline of distributed agent software consists of computational entities which interact with one another towards a common goal that is beyond their individual capabilities. There are many varying definitions of the notion of software agent. The main common haracteristics are: (a) the possession of individual goals, resources and competences, (b) the ability to perceive and to act, to some degree, on the near environment; this includes communications with other agents, (c) the faculty to perform actions with some level of autonomy and/or reactiveness, and eventually to replicate, (d) and the capacity to provide services. The above-mentioned properties induce that the agent paradigm is also very suitable for building adaptive applications, where interactions amongst the different entities involved may be altered during the course of computation, and where this change must have an impact on the software behaviour. This is all the more important in scalable systems because environment characteristics can vary a lot from one vicinity of the network to another. However, as the amount of locations and agents within the system increases, so does the probability of failures. As mentioned earlier, multi-agent applications rely on collaboration amongst agents. It follows that the failure of one of the involved agents can bring the whole computation to a dead end. It is therefore crucial to apply dependability protocols when distributing agent systems over large-scale networks. Replicating agents seems to be the only efficient way to achieve fault tolerance in scalable agent systems. A replicated software component is defined as a software component that possesses a representation on two or more hosts. Consistency between replicas can be maintained following two main strategies: the active one in which all replicas process all input messages concurrently, and the passive one in which only one of the replicas processes all input messages and periodically transmits its current state to the other replicas. Each type of strategy has its advantages and disadvantages. Active replication is dedicated to applications which require full recovery over short delays. Passive replication makes for lower overhead in a failure-free environment but recovery is less efficient. The choice of the most suitable strategy is directly dependent of the environment context, especially the failure rate, the kind of failure that must be tolerated, and the application requirements in terms of recovery delay and

overhead. In practice, replicating every agent over a large-scale network would lead to excessive overheads, both in terms of network load and in terms of computation time. Besides, at any given point of the computation, only a small subset of agents is likely to be crucial to its continuation. Therefore only the specific agents which are temporarily identified as crucial to the application should be replicated, and the applied strategy should be carefully selected. Our work serves a twofold objective: (1) to provide efficient fault-tolerance to multi-agent systems through selective agent replication, (2) to take advantage of the specificities of multi-agent platforms to develop a suitable architecture for performing adaptive fault-tolerance within distributed applications; such applications would then be liable to operate efficiently over large-scale networks. Our project has lead to the design and development of DARX, an architecture for fault-tolerant agent computing. As opposed to the main conventional distributed programming architectures, ours offers dynamic properties: software elements can be replicated and unreplicated on the spot and it is possible to change the current replication strategies on the fly. More importantly, DARX allows to automate its fault tolerance support for agent applications with respect to the behaviour of their runtime environment. The originality of our approach lies in two features: (i) an automated replication service which chooses for the application which of its computational components are to be made dependable, to which degree, and at what point of the execution, and (ii) the hierarchic architecture of the middleware which ought to provide suitable support for large-scale applications. DARX consists of several services. A failure detection service maintains dynamic lists of all the running hosts as well as of the valid software elements which participate to the supported application, and notifies the latter of suspected failure occurrences. A naming and localisation service generates a unique identifier for every agent in the system, and returns the addresses for all agent replicas in response to an agent localisation request. A system observation service monitors the behaviour of the underlying distributed system: it collects low-level data by means of OS-compliant probes and diffuses processed trace information so as to make it available for the adaptive replication control process. An application analysis service builds a global representation of the supported agent application in terms of fault tolerance requirements. A replication service brings all the necessary mechanisms for replicating agents, maintaining the consistency between replicas of a same agent, and automating replication scheme adaptation for every agent according to the data gathered through system monitoring and application analysis. An interfacing service offers wrapper-making solutions for agents, thus rendering the DARX middleware usable by various multi-agent systems and even making it possible to introduce interoperability amongst different systems.

Our current undergoing research addresses the following issues:

- Scalable monitoring: we are looking into how to extract and aggregate useful information about the computing environment, and how to diffuse this information in a timely manner to the locations that require it. A small prototype has been built using JNI and JAX, and now awaits deployment and performance measurement in order to gain further knowledge of the time constraints that impair monitoring in a scalable environment.
- Fault-tolerance by contract: we are working on the establishment of contracts between DARX and its supported applications that satisfy both the application requirements and the environment constraints. This work also looks into the distributed handling of exceptions raised when transparent replication cannot be guaranteed anymore at some point of the computation. Two projects have stemmed from this axis of research:
    1. FTATC (Towards Fault-Tolerant Cooperative Air Traffic Management), a 1-year project financed by EuroControl and the European Community. http://www-src.lip6.fr/homepages/facoma.officiel/FTATC/
    2. FACOMA (Fiabilisation Adaptative d'applications COopératives Multi-Agents), a 3-year project supported in the context of the ANR-Setin frame.
- Fault tolerance in mobility: we are designing a middleware for the deployment of distributed algorithms among mobile devices. The originality of our approach is double: (a) we view partial and total disconnexion as types of failures and aim to integrate fault tolerance solutions in order to guarantee the continuity of the computation in such a context, and (b) we provide a modeling

language which is close to Pi-calculus and yet focuses on communication channels in order to represent replicated applications and introduce failures. This work is part of FRAME, a project funded by the LIP6. As of October 1st, 2007, there is a PhD student working full time on this project on behalf of the REGAL team.

7. Other Grants and Activities

7.1. National initiatives

7.1.1. Facoma - (2007–2009)

Members: LIP6, LIRMM, Regal

Funding: Facoma project is funded by ANR SETIN

Objectives: The fault tolerance research community has developped solutions (algorithms and architectures), mostly based on the concept of replication, applied for instance to data bases. But, these techniques are almost always applied explicitely and statically. This is the responsability of the designer of the application to identify explicitly which critical servers should be made robust and also to decide which strategies (active or passive replication) and their configurations (how many replicas, their placement). Meanwhile, regarding new cooperative applications, which are very dynamic, for instance: decision support systems, distributed control, electronic commerce, crisis management systems, and intelligent sensors networks, - such applications increasingly modeled as a set of cooperative agents (multi-agent systems) -, it is very difficult, or even impossible, to identify in advance the most critical agents of the application. This is because the roles and relative importances of the agents can greatly vary during the course of computation, interaction and cooperation, the agents being able to change roles, strategies, plans, and new agents may also join or leave the application (open system). Our approach is in consequence to give the capacity to the multi-agent system itself to dynamically identify the most critical agents and to decide which abilisation strategies to apply to them.

7.1.2. Gedeon - (2004–2007)

Members: IMAG-ID, IMAG-LSR, IBCP, Regal

Funding: Gedeon project is funded by ACI MD

Objectives: File systems (FS) are commonly used to store data. Especially, they are intensively used in the community of large scientific computing (astronomy, biology, weather prediction) which needs the storage of large amouts of data in a distributed manner. In a GRID context (cluster of clusters), traditional distributed file systems have been adapted to manage a large number of hosts (like the Andrew File System). However, such file systems remain inadequate to manage huge data. They are suited for traditional unix (small) files. Thus, the grain of distribution is typically an entire file and not a piece of file which is essential for large files. Furthermore, the tools for managing data (e.g, interrogation, duplication, consistency) are unsuited for large sizes.

Database Management Systems (DBMS) provides different abstraction layers, high level languages for data interrogation and manipulation etc. However, the imposed data structuration, the low distribution, and the usually monolithic architecture of DBMSs limit their utilisation in the scientific computing context.

The main idea of the Gedeon project is to merge the functions of file systems and DBMS, focusing on structuration of meta-data, duplication and coherency control. Our goal is NOT to build a DBMS describing a set of files. We will study how database management services can be used to improve the efficiency of file access and to increase the functionality provided to scientific programmers.

7.1.3. Respire - (2005–2008)

Members: LIP6, Atlas (IRISA), Paris (IRISA), Regal

Funding: RESPIRE project is funded by ANR (ARA MDSA)

Objectives: The Respire project aims to develop support for sharing information (including either data or meta-data) and services for managing this information in a Peer-to-Peer (P2P) environment. A P2P architecture is potentially more scalable than previous client-server approaches, but raises many interesting scientific issues. Peers are autonomous and may join or leave the network at any time. Peers publish resources for sharing, such as data or services, and may use resources published by other peers. Users may collaborate without any explicit or implicit hierarchy. We target applications that enable world-wide spread professional communities (such as a group of researchers) to collaborate, or learning scenarios. These applications manipulate heterogenous, semantically rich data. Therefore they require more advanced functionality than existing P2P file systems.

Respire is a collaboration between research teams from different areas, distributed databases and distributed systems, which have until now largely ignored each other. This synergetic approach enables each community to question hidden assumptions, and to take into account new approaches and requirements.

7.1.4. PlayAll - (2007–2009)

Members: PME : Darkwoks, Atonce, Bionatics, Fandango Games, Load Inc, Kilotonn, Sixela, SpirOps, Voxler, White Birds, Wizrbox - Public : CNAM, ENST, ENJMIN, LIP6 (REgal), LIRIS

Funding: PLAYALL project is funded by Pôle de Compitivité - Cap Digital

Objectives: The goal is the build a middleware adapted to the different game platforms (Sony Play Station 3, Nitendo DS, Wii, Xbox, PC). The contribution of Regal concerns distributed algorithms taking into account QoS contraints.

7.1.5. Recall - (2005–2008)

Members: LIP6, Atlas (IRISA), Paris (IRISA), Regal

Funding: Recall is funded by INRIA (Action de Recherche Coopérative)

Objectives: Recall aims to develop optimistic replication algorithms for supporting massive collaborative editing applications. The goal is to enable classical collaborative applications to scale and to tolerate faults, by deploying them above peer-to-peer networks, and without expensive hardware requirements. This project will show that P2P networks are a viable solution, not only for distributing content, but also for creating and editing it.

7.1.6. Fracas - (2007–2009)

Members: ARES (Rhones-Alpes), DIONYSOS (IRISA), Grand-Large (Futurs), Regal(Paris-Rocquencourt)

Funding: Fracas is funded by INRIA (Action de Recherche Coopérative)

Objectives: We propose to define a new middleware dedicated for sensor networks. This middleware must tolerate failures and specific attacks these networks are subject.

7.2. European initiatives

7.2.1. Grant from Microsoft Research Cambridge

Data replication enables cooperative work, improves access latency to data shared through the network, and improves availability in the presence of failures. This grant supports a doctoral student for studying consistency

between replicas of mutable, semantically-rich data in a peer-to-peer fashion. This study should enable to engineer distributed systems and applications based on them, supporting cooperative applications in large-scale collaboration networks. It includes a systematic exploration of the solution space, in order to expose the cost vs. performance vs. availability vs. quality trade-offs, and understanding fault tolerance and recovery aspects. This work combines formal approaches, simulation, implementation, and measurement.

7.2.2. Grid4All - (2006-2009)

Members: France Télécom Recherche et Développement, INRIA (Regal, Atlas and Grand-Large), SICS, KTH, ICCS, UPRC, UPC, Redidia.

Funding: European Commission, 6th Framework Programme, STREP (Specific Targeted Research Project)

Objectives: Grid4All embraces the vision of a "democratic" Grid as a ubiquitous utility whereby domestic users, small organizations and enterprises may draw on resources on the Internet without having to individually invest and manage computing and IT resources. This project is funded by the 6th Framework Programme of the European Commission. It involves institutional and industrial partners. Its budget is slightly over 4.8 million euros.

Grid4All has the following objectives:

- To alleviate administration and management of large scale distributed IT infrastructure, by pioneering the application of component based management architectures to self-organizing peer-to-peer overlay services.
- To provide self-management capabilities, to improve scalability, resilience to failures and volatility thus paving the way to mature solutions enabling deployment of Grids on the wide Internet.
- To widen the scope of Grid technologies by enabling on-demand creation and maintenance of dynamically evolving scalable virtual organisations even short lived.
- To apply advanced application frameworks for collaborative data sharing applications executing in dynamic environments.
- To capitalize on Grids as revenue generating sources to implement utility models of computing but using resources on the Internet.

Grid4All will help to bring global computing to the broader society beyond that of academia and large enterprises by providing an opportunity to small organisations and individuals to reap the cost benefit of resource sharing without however the burdens of management, security, and administration.

The consortium will demonstrate this by applying Grid4All in two different application domains: collaborative tools for e-learning targeting schools and digital content processing applications targeting residential users.

7.3. International initiatives

JAIST (Japon). With the group of Prof. Xavier Defago we investigate various aspects of self-organization and fault tolerance in the context of robots networks.

UNLV (SUA) With the group of Prof. Ajoy Datta we collaborate in designing self* solutions for the computations of connected covers of query regions in sensor networks.

Technion (Israel). We collaborate with Prof. Roy Friedman on divers aspects of dynamic systems ranging from the computation of connected covers to the design of agreement problems adequate for P2P networks.

COFECUB (Brazil). With the group of Prof. F. Greve. (Univ. Federal of Bahia), we investigate various aspects of failure detection for dynamic environement such as MANET of P2P systems.

CONYCIT (Chili). In 2007, we start on new collaboration with the group of X. Bonnaire Fabre (Universidad Técnica Federico Santa María - Valparaiso). The main goal is to implement trusted services in P2P environment. Even if it is near impossible to fully trust a node in a P2P system, managing a set of the most trusted nodes in the system can help to implement more trusted and reliable services. Using these nodes, can reduce the probability to have some malicious nodes that will not correctly provide the given service. The project will have the following objectives: 1. To design a distributed membership algorithm for structured Peer to Peer networks in order to build a group of trusted nodes. 2. To design a maintenance algorithm to periodically clean the trusted group so as to avoid nodes whose reputation has decreased under the minimum value. 3. To provide a way for a given node X to find at least one trusted node. 4. To design a prototype of an information system, such as a news dissemination system, that relies on the trusted group.

Informal collaboration with INESC. João. Barreto, a PhD student at INESC and Instituto Superior Técnico spent six months in Projet Regal, working on ubiquitous information sharing in mobile ad-hoc networks. He took an important role in the design of our decentralised commitment algorithm .

8. Dissemination

8.1. Program committees and responsibilities

Luciana Arantes is:

- Member of the program committee of the 6ème Conférence française sur les systèmes d'exploitation, CFSE-6, Friburg, Switzerland, february 2008.
- Member of the program committee of Workshop de Sistemas Operacionais, Brasil 2007.
- Member of the "Commission de spécialiste" of the Paris 5 University.

Bertil Folliot is:

- Member of the program committee of 6th International Symposium on Parallel and Distributed Computing, Hagenberg, Austria, july 2007.
- Member of the program committee of the 6ème Conférence française sur les systèmes d'exploitation, CFSE-6, Friburg, Switzerland, february 2008.
- Elected member of the Commission de spécialistes of the Paris 6 University.
- Co-chair of the middleware group of GdR ASR (Hardware, System and Network).
- Elected member of the IFIP WG10.3 working group (International Federation for Information Processing - Concurrent systems).
- Member of the « Advisory Board » of EuroPar (International European Conference on Parallel and Distributed Computing), IFIP/ACM.
- Member of the « steering committee of the International Symposium on Parallel and Distributed Computing".
- Reviewer for the IET Software Journal.

Maria Gradinariu is:

- Member of the program committee Autonomics 2007 (International Conference on Autonomic Computing and Communication Systems)
- Member of the program committee DASSON 2007 (Workshop on Dependable Application Support for Self-Organizing Networks)
- Member of the program committee SSS 2007 (Symposium on Self-stabilizing Systems)
- Member of the program committee ISORC 2008 (International Symposium Object/component/service oriented real-time distributed computing)
- Member of the program committee Algotel 2008 (10eme rencontres francophones sur les aspects algorithmiques de telecommunications)

Mesaac Makpangou is:

- Member of the “Commission de spécialistes” of the University of Marne La Vallée.

Olivier Marin is:

- Member of the board of Distributed Systems Online (DSO)
- Community editor for DSO Distributed Agents

Pierre Sens is:

- Chair of program committee of the 6ème Conférence française sur les systèmes d’exploitation, CFSE-6, Friburg, Switzerland, february 2008.
- Member of the program committee of Parallel Computing Journal 2007 - Special Issue of Parallel Computing Journal on Large Scale Grid, (PARCO’07)
- Member of the program committee of Second International Workshop on Distributed Autonomous Network Management Systems, (DANMS’07).
- Member of the program committee of SSS’2007 - *9th International Symposium on Stabilization, Safety, and Security of Distributed Systems (formerly Symposium on Self-stabilizing Systems) (SSS 2006) November 17th-19th, 2006, Dallas, Texas, USA*
- Member of the “Commission de spécialistes” of ENS-Cachan and Paris 11 - Orsay University.
- Elected member of the “Institut de Formation Doctorale” of Paris 6 University.
- Vice-chair of the LIP6 laboratory.
- Reviewer for JPDC and TPCDS journals

Marc Shapiro is:

- Member of the PC European Computer Science Summit 2007, Berlin October 2007
- proposal reviewer for Swiss National Science Foundation
- member of PC for ICDCS 2007 (Int. Conf. on Distributed Computing Systems).
- member of PC for PPoPP 2008 (Principles and Practice of Parallel Programming)
- Invited speaker at ETH Colloquium, ETHZ, Zurich, Switzerland
- member of PC of OPODIS 2007 (International Conference On Principles Of Distributed Systems)
- reviewer for Springer Distributed Computing
- PC member, VLDB workshop on Data Management for Grids
- reviewer for Swedish Research Council (Vetenskapsrådet)
- reviewer for ICDE
- reviewer for Transactions on Parallel and Distributed Systems
- PC chair, topic "Distributed algorithms and systems" at Europar 2008
- organiser of Workshop on Decentralised Mechanisms for User Communities at EuroSys 2008
- Member of Steering Committee of EuroSys conference
- invited to Monte Verità seminar "30 Year Perspective on Replication", nov. 2007
- invited to Dagstuhl seminar "Transactional Memory: From Implementation to Application", June 2008

8.2. PhD reviews

Bertil Folliot was PhD rewiever of:

- Ali Özcan. Conception et implantation d'un environnement de développement de logiciels à base de composants : Applications aux systèmes multiprocesseurs sur puce. Thèse de Doctorat de l'Institut National Polytechnique de Grenoble (directeur : Jean-Bernard Stefani), Grenoble, mars 2007.
- Yann Hodique. Sûreté et optimiusation par les systèmes de types ouvert et contraint. Thèse de Doctorat de lUniversité de Lille I (directeur : Gilles Grimaud, Isabelle Simplot-Ryl, Mireille Clerbout), Lille, avril 2007.
- Jan Lemeire. Learning Causal Models of Multivariate Systems and the Value of its Performance Modeling of Computer Programs. PhD. Dissertation Vrije Universiteit Brussel (directeur : Eric Dirkx), Bruxelles, novembre 2007.

Pierre Sens was PhD reviewer of:

- Ludovic Courtès. Sauvegarde coopérative de données pour dispositifs mobiles. Thèse de Doctorat de l'Insitut Polytechnique de Toulouse (directeur : D. Powell / LAAS).
- Emmanuel Jeanvoine. Intergiciel pour l'exécution efficace et fiable d'applications distribuées dans les grilles dynamiques de très grande taille. Thèse de Doctorat de l'Université Rennes 1 (directrice : C. Morin / IRISA).
- Etienne Rivière. Réseaux logiques collaboratifs pour la recherche décentralisée dans les systèmes à large échelle. Thèse de Doctorat de l'Université Rennes 1 (directrice : A.-M. Kermerrac / IRISA).
- Christian Delbé. Tolérance aux pannes pour objets actifs asynchrones : protocole, modèle et expérimentation. Université de Nice - Sophia Antipolis (directeur : D. Caromel)

8.3. Teaching

- Luciana Arantes
  - Principles of operating systems in Licence d'Informatique, Université Paris 6
  - Operating systems kernel in Maîtrise d'Informatique, Université Paris 6
  - Responsible for projects in operating system, Maîtrise d'Informatique, Université Paris 6
- Bertil Folliot
  - Principles of operating systems in Licence d'Informatique, Université Paris 6
  - Distributed algorithms and systems in Master Informatique, Université Paris 6
  - Distributed systems and client/serveur in Master Informatique, Université Paris 6
  - Projects in distributed programming in Master Informatique, Université Paris 6
- Mesaac Makpangou
  - Systems and networks, Master, Pôle Universitaire Leonard de Vinci
- Oliver Marin
  - Operating system programming, Master d'Informatique, Université Paris 6
  - Operating system Principles, Licence d'Informatique, Université Paris 6
  - Parallel and distributed systems, Master d'Informatique, Université Paris 6
  - Client/server arcitecture, Licence professionelle d'Informatique, Université Paris 6
- Pierre Sens
  - Principles of operating systems in Licence d'Informatique, Université Paris 6

  - Operating systems kernel in Master Informatique, Université Paris 6
  - Distributed systems and algorithms in Master Informatique, Université Paris 6
- Gaël Thomas
  - Principles of operating systems in Licence d'Informatique, Université Paris 6
  - Operating systems kernel in Master Informatique, Université Paris 6
  - Distributed systems and Middleware in Master Informatique, Université Paris 6

9. Bibliography

Major publications by the team in recent years

[1] L. ARANTES, D. POITRENAUD, P. SENS, B. FOLLIOT. *The Barrier-Lock Clock: A Scalable Synchronization-Oriented Logical Clock*, in "Parallel Processing Letters", vol. 11, n^{o} 1, 2001, p. 65–76.

[2] M. BERTIER, L. ARANTES, P. SENS. *Hierarchical token based mutual exclusion algorithms*, in "Proceedings of the 4th IEEE/ACM International Symposium on Clu ster Computing and the Grid (CCGrid '04), Chicago (USA)", IEEE Society Press, April 2004.

[3] M. BERTIER, L. ARANTES, P. SENS. *Distributed Mutual Exclusion Algorithms for Grid Applications: A Hierarchical Approach*, in "JPDC: Journal of Parallel and Distributed Computing", vol. 66, 2006, p. 128–144.

[4] M. BERTIER, O. MARIN, P. SENS. *Implementation and performance of an adaptable failure detecto r*, in "Proceedings of the International Conference on Dependable Systems and Networks (DSN '02)", June 2002.

[5] M. BERTIER, O. MARIN, P. SENS. *Performance Analysis of Hierarchical Failure Detector*, in "Proceedings of the International Conference on Dependable Systems and Networks (DSN '03), San-Francisco (USA)", IEEE Society Press, June 2003.

[6] J.-M. BUSCA, F. PICCONI, P. SENS. *Pastis: a Highly-Scalabel Multi-User Peer-to-Peer File Systems*, in "Euro-Par'05 - Parallel Processing, Lisboa, Portugal", Lecture Notes in Computer Science, Springer-Verlag, August 2005.

[7] A.-M. KERMARREC, A. ROWSTRON, M. SHAPIRO, P. DRUSCHEL. *The IceCube approach to the reconciliation of divergent replicas*, in "20th Symp. on Principles of Dist. Comp. (PODC), Newport RI (USA)", ACM SIGACT-SIGOPS, August 2001, http://research.microsoft.com/research/camdis/Publis/podc2001.pdf.

[8] N. KRISHNA, M. SHAPIRO, K. BHARGAVAN. *Brief announcement: Exploring the Consistency Problem Space*, in "Symp. on Prin. of Dist. Computing (PODC), Las Vegas, Nevada, USA", ACM SIGACT-SIGOPS, July 2005.

[9] O. MARIN, M. BERTIER, P. SENS. *DARX - A Framework For The Fault-Tolerant Support Of Agent S oftware*, in "Proceedings of the 14th IEEE International Symposium on Sofwat are Reliability Engineering (ISSRE '03), Denver (USA)", IEEE Society Press, November 2003.

[10] F. OGEL, G. THOMAS, A. GALLAND, B. FOLLIOT. *MVV : une Plate-forme à Composants Dynamiquement Reconfigurables — La Machine Virtuelle Virtuelle*, 2004.

Year Publications

Doctoral dissertations and Habilitation theses

[11] J.-M. BUSCA. *Pastis : un système pair à pair de gestion de fichiers*, Ph. D. Thesis, UniversitéPierre et Marie Curie (Paris 6), 4, place Jussieu, Paris, december 2007.

[12] I. CHABBOUH. *FRACS: Un système de fragmentation et de distribution d'applications Web*, Ph. D. Thesis, Université Pierre et Marie Curie (Paris 6), 4, place Jussieu, Paris, december 2007.

Articles in refereed journals and book chapters

[13] E. ANCEAUME, R. FRIEDMAN, M. GRADINARIU. *Managed Agreement: Generalizing two fundamental distributed agreement problems*, in "Inf. Process. Lett.", vol. 101, n^{o} 5, 2007, p. 190-198.

[14] J. BEAUQUIER, M. GRADINARIU, C. JOHNEN. *Randomized self-stabilizing and space optimal leader election under arbitrary scheduler on rings*, in "Distributed Computing", vol. 20, n^{o} 1, 2007, p. 75-93.

[15] Z. GUESSOUM, J.-P. BRIOT, N. FACI, O. MARIN. *Towards Reliable Multi-Agent Systems - An Adaptive Replication Mechanism*, in "Multiagent and Grid Systems", 2007.

[16] R. MORALES, S. MONNET, G. ANTONIU, I. GUPTA. *MOve: Design and Evaluation of A Malleable Overlay for Group-Based Applications*, in "IEEE TNSM Special Issue on Self-Management", 2007.

Publications in Conferences and Workshops

[17] A. D. L. ALMEIDA, J.-P. BRIOT, S. AKNINE, Z. GUESSOUM, O. MARIN. *Towards autonomic fault-tolerant multi-agent systems*, in "2nd Latin American Autonomic Computing Symposium (LAACS'07), Petropolis, Brazil", 2007.

[18] J. BARRETO, P. FERREIRA, M. SHAPIRO. *Exploiting our computational surroundings for better mobile collaboration*, in "Int. Conf. on Mobile Data Management (MDM'07), Mannheim, Germany", May 2007, p. 110–117, http://www-sor.inria.fr/~shapiro/papers/DecoupledConsistency-MDM-2007.pdf.

[19] S. BIANCHI, A. K. DATTA, P. FELBER, M. GRADINARIU. *Stabilizing Peer-to-Peer Spatial Filters*, in "ICDCS", 2007, 27.

[20] S. BIANCHI, P. FELBER, M. GRADINARIU. *Content-Based Publish/Subscribe Using Distributed R-Trees*, in "Euro-Par", 2007, p. 537-548.

[21] L. BLIN, M. G. POTOP-BUTUCARU, S. TIXEUIL. *On the Self-stabilization of Mobile Robots in Graphs*, in "Opodis", 2007.

[22] X. BONNAIRE, O. MARIN. *Recursive Replication: A Survival Solution for Structured P2P Information Systems to Denial of Service Attacks*, in "1st International Workshop on Peer to Peer Networks (PPN´07)", vol. 2, OTM Workshops, November 2007, p. 931–940.

[23] D. CANEPA, M. G. POTOP-BUTUCARU. *Stabilizing Flocking Via Leader Election in Robot Networks*, in "SSS", 2007, p. 52-66.

[24] A. K. DATTA, M. G. POTOP-BUTUCARU, R. PATEL, A. YAMAZAKI. *Self* Minimum Connected Covers of Query Regions in Sensor Networks*, in "SSS", 2007, p. 204-218.

[25] M. GRADINARIU, S. TIXEUIL. *Conflict Managers for Self-stabilization without Fairness Assumption*, in "ICDCS", 2007, 46.

[26] C. HERAULT, G. THOMAS, P. LALANDA. *A distributed service-oriented mediation tool*, in "Proceedings of IEEE International Conference on Services Computing (SCC 2007), Salt Lake City, Utah, USA", July 2007, p. 403-409.

[27] C.-L. IGNAT, G. OSTER, P. MOLLI, M. CART, J. FERRIÉ, A.-M. KERMARREC, P. SUTRA, M. SHAPIRO, L. BENMOUFFOK, J.-M. BUSCA, R. GUERRAOUI. *A Comparison of Optimistic Approaches to Collaborative Editing of Wiki Pages*, in "Int. Conf. on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), White Plains, NY, USA", n^{o} 3, November 2007.

[28] F. PICCONI, B. BAYNAT, P. SENS. *An analytical estimation of durability in DHTs*, in "Proceedings of 4th International Conference on Distributed Computing and Internet Technology (ICDCIT 20007), Bangalore, India", Springer Verlag, December 2007.

[29] F. PICCONI, B. BAYNAT, P. SENS. *Predicting durability in DHTs using Markov chains*, in "Proceedings of International Workshop on Advanced Storage Systems (ADSS 2007), Lyon, France", IEEE Computer Society, October 2007.

[30] J. SOPENA, F. LEGOND, L. ARANTES, P. SENS. *A Composition Approach to Mutual Exclusion Algorithms for Grid Applications*, in "The 36th International Conference on Parallel Processing (ICPP07 - XiAn, CN), Los Alamitos, CA, USA", vol. 0, IEEE Computer Society, September 2007, 65.

[31] P. SUTRA, J. BARRETO, M. SHAPIRO. *Decentralised Commitment for Optimistic Semantic Replication*, in "Int. Conf. on Coop. Info. Sys. (CoopIS), Vilamoura, Algarve, Portugal", November 2007, http://www-sor.inria.fr/~shapiro/papers/sutra-barreto-shapiro-coopis40.pdf.

[32] P. SUTRA, M. SHAPIRO. *Comparing Optimistic Database Replication Techniques*, in "Bases de Données Avancées, Marseille, France", October 2007, http://www-sor.inria.fr/~shapiro/papers/sutra-shapiro-bda2007.pdf.

Internal Reports

[33] D. CONAN, P. SENS, L. ARANTES, M. BOUILLAGUET. *Failure, Disconnection and Partition Detection in Mobile Environment*, Research Report, n^{o} 6184, INRIA, 05 2007, https://hal.inria.fr/inria-00144801.

[34] C. IGNAT, G. OSTER, P. MOLLI, M. CART, J. FERRIÉ, A.-M. KERMARREC, P. SUTRA, M. SHAPIRO, L. BENMOUFFOK, J.-M. BUSCA, R. GUERRAOUI. *A Comparison of Optimistic Approaches to Collaborative Editing of Wiki Pages*, Research Report, n^{o} RR-6278, INRIA, September 2007, https://hal.inria.fr/inria-00169395.

[35] F. PICCONI, J.-M. BUSCA, P. SENS. *An experimental evaluation of the Pastis peer-to-peer file system under churn*, Research Report, n^{o} 6114, INRIA, 02 2007, https://hal.inria.fr/inria-00128869.

[36] P. SENS, L. ARANTES, M. BOUILLAGUET. *Asynchronous Implementation of Failure Detectors with partial connectivity and unknown participants*, Research Report, n^{o} 6088, INRIA, 01 2007, https://hal.inria.fr/inria-00122517.

[37] M. SHAPIRO, N. PREGUIÇA. *Designing a commutative replicated data type*, Rapport de recherche, n^{o} RR-6320, INRIA, Rocquencourt, October 2007, http://hal.inria.fr/inria-00177693/.