INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Team Regal

Resource management in large scale distributed systems

Rocquencourt

THEME COM

Activity Report

2005

Table of contents

1. Team

Regal is a joint team between INRIA, CNRS and Paris 6 University, through the "Laboratoire d'Informatique de Paris 6", LIP6 *(UMR 7606).*

Head of project-team
Pierre Sens [Professor University of Paris 6]

Vice-head of project-team
Mesaac Makpangou [Research associate (CR) INRIA]

Administrative assistant
Nicole Loza [ITA]

Staff member INRIA
Marc Shapiro [Research Director (DR) INRIA]

Staff member LIP6
Luciana Arantes [Associate professor University of Paris 6]
Bertil Folliot [Professor Univeristy of Paris 6]
Oliver Marin [Associate professor Univeristy of Paris 6]

Postdoct
Guillaume Vauvert [INRIA Post doctoral research fellow (until october 2005)]

Ph. D. student
Jean-Michel Busca [INRIA until august, ATER University of Paris since September]
Baptiste Canton [Microsoft research grant - University of Paris 6]
Ikram Chabbouh [University of Tunis - INRIA]
Charles Clement [University of Paris 6]
Nicolas Geoffray [University of Paris 6]
Corina Ferdean [INRIA]
Nicolas Gibelin [INRIA]
Assia Hachichi [University of Paris 6]
Cyril Martin [University of Paris 6]
Fabio Piconni [University of Paris 6]
Julien Sopena [University of Paris 6]
Pierre Sutra [INRIA]

2. Overall Objectives

2.1. Overall Objectives

Regal is a joint research team between LIP6 and INRIA-Rocquencourt. Regal focuses on management of large scale distributed systems (especially P2P and Grid architectures). A significant axis of the project work is devoted to large scale replication for applications which have strong constraints in terms of dynamicity (multi-agents applications, resource servers on the Web).

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 :

- Continuous failures occurrence: 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 problem is mainly due to the Fischer, Lynch, Paterson impossibility which 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 a large scale context like peer-to-peer overlay networks, 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, 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.

5. Software

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

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

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.

6. New Results

6.1. Introduction

In 2005, we focus our research on the following areas :

- distributed algorithms for Grid configuration,
- impact of churn is peer-to-peer data storage,
- dynamic adaptation of virtual machines,
- services management in large scale environments,
- theorical aspects of replication strategies,
- dynamic replication in distributed multi-agent systems.

6.2. Distributed algorithms

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

Our current research focuses on adapting distributed algorithms to large-scale and heterogeneous environments. We target particularly two basic blocks : mutual exclusion algorithms 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. Exploiting these points, we have proposed two distributed mutual exclusion algorithms, based on Naimi-Trehel's token-based algorithm: the first on takes into account latency gaps, especially those between local and remote clusters of machines in a Grid environment while the second one provides scalable fault tolerance feature by minimizing the use of broadcast support. It exploits the distributed queue of token requests to offer fault tolerance support. Both algorithms have been published in recent conferences [31], [2] and the first one is published in the french national conference on operating systems [16] and in a JPDC journal [10].
- 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 propose a new scalable implementation of failure detector (called GFD for *Grid Failure Detector*) taking into account the network topology. We target grid configurations typically consisting of federation of clusters. Our implementation is a variant of the heartbeat detector which is adaptable and can support scalable applications. Our detector has the following properties:
  - **scalability:** order to adapt our detector to large configurations, we define a hierarchical version. Failure detection is handled at two different levels. At cluster-level, each node sends heartbeats to all the other nodes of its own cluster. Each cluster selects a mandatory, which is in charge of handling failure detection at grid level. Note that this mandatory may fail: in this case, its failure is detected at cluster-level, and another mandatory is selected.

– **adaptability:** The implementation is based on a basic failure detection layer introduced in [3], [4]. This layer is a variant of the heartbeat detector: it adapts the sending period of *"I am alive"* messages as a function of the network quality of service and of the application requirements. For each application, an adaptor is plugged onto the failure detector layer. Its aim is to adapt the quality of service provided by the basic layer according to application needs. This architecture guarantees the required quality of service and avoids network overload sharing the detection service between several applications.

GFD is now included in the JuxMem framework [9] developped by the Paris Team.

6.3. Impact of churn in peer-to-peer data storage

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

Since 2003, we develop Pastis [17] 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. This is very common in existing peer-to-peer file sharing networks, as many users connect to the overlay to download a particular file, and disconnect soon after the download has finished. Although intermittent connections are not particularly harmful in file sharing networks, this kind of unstable user behavior is undesirable on DHTs. Contrary to file sharing systems, DHTs are designed to guarantee data persistence. This is achieved by replicating data blocks on geographically dispersed nodes, which minimizes the probability of correlated failures, and by regenerating replicas as soon as they leave the network so that the replication factor is kept constant. This reduces the risk of data becoming unavailable if all replicas leave the network, but it also means that as nodes join and leave the network the DHT maintenance algorithm needs to transfer a large number of replicas from one node to another, consuming a lot of bandwidth. Furthermore, DHTs clients lack any flexibility to choose where their data is stored in the overlay.

In 2005 we study the effect of churn on Pastis and more generally on DHT. We propose a new incentives-based mechanism to increase the availability of DHT nodes, thereby providing better data persistence for DHT users. High availability increases a node's reputation, which translates into access to more DHT resources and a better Quality-ofService. The mechanism required for tracking a node's reputation is completely decentralized, and is based on certificates reporting a node's availability which are generated and signed by the node's neighbors. An audit mechanism deters collusive neighbors from generating fake certificates to take advantage of the system. Prelimary results of this proposal will be published in the HotP2P workshop [28].

6.4. Virtual virtual machine (VVM)

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

The VVM group works on flexible execution environments, based on a HAL (Hardware Abstraction Layer) and a flexible dynamic compiler, free of any predefined, imposed abstractions.

In 2005, VVM group focuses on the following themes:

Adaptive language for middleware: Today, component oriented middlewares are used to design, develop and deploy distributed applications easily. They ensure the heterogeneity, interoperability, and reuse of software modules. Several standards address this issue: CCM (CORBA Component Model), EJB (Enterprise Java Beans) and .Net. However they offer a limited and fixed number of system services, and their deployment and configuration mechanisms cannot be used by any language nor API dynamically. As a solution, we propose a generic high-level language to adapt system services dynamically in existing middlewares. This solution is based on a highly adaptable platform which enforces adaptive behaviours, and offers a means to specify and adapt system services dynamically. A first prototype was achieved for the OpenCCM platform, and good performances were obtained.

Dynamic bytecode offloading in a network of workstations: We propose a new runtime infrastructure able to transparently distribute computation between interconnected workstations. Application source code is not modified: instead, dynamic aspect weaving within an extended virtual machine allows to monitor and distribute entities dynamically. Applications that benefit from our system can be (i) applications that execute on resource constrained devices, and will offload to nearby computers, or (ii) large scale applications, where dynamic load-balancing during high load will offer a guarantee of service. The platform developed is an extension of a standard and generic Java virtual machine, the JNJVM, that provides abstractions to modify on the fly the behavior of the runtime. Our offloading system is thus loaded only when the application needs it. Existing applications can be used, and the offloading system can be deployed after launching time, without interrupting the application, even if the administrator has not planned to inject this non functional behavior. A first evaluation of the offloading system shows that our technique can absorb an overload in a web server.

FlexORB: Smart devices, such as personal assistants, mobile phone or smart cards, continuously spread and thus challenge every aspect of our lives. However, such environments exhibit specific constraints, such as mobility, high-level of dynamism and most often restricted resources. Traditional middlewares were not designed for such constraints and, because of their monolithic, static and rigid architectures, are not likely to become a fit. In response, we propose a flexible micro-ORB, called FlexORB, that supports on demand export of services as well as their dynamic deployment and reconfiguration. FlexORB supports mobile code through an intermediate code representation. It is built on top of Nevermind, a flexible minimal execution environment, which uses a reflexive dynamic compiler as a central common language substrate upon which to achieve interoperability. Preliminary performance measurements show that, while being relatively small (120 KB) and dynamically adaptable, FlexORB outperforms traditional middlewares such as RPC, CORBA and Java RMI.

Dynamic aspect support in the VVM As systems grow more and more complex, raising severe evolutions and management difficulties, computational reflection and aspect-orientation have proven to enforce separation of concerns principles and thus to address those issues. However, most of the existing solutions rely either on a static source code manipulation or on the introduction of extra-code (and overhead) to support dynamic adaptation. Whereas those approaches represent the extreme of a specter, developers are left with this rigid tradeoff between performance and dynamism. A first step toward a solution was the introduction of specialized virtual machines to support dynamic aspects into the core of the execution engine. However, using such dedicated runtimes limits applications' portability and interoperability. In order to reconcile dynamism and performance without introducing portability and interoperability issues, we propose a dynamic reflexive runtime that uses reflection and dynamic compilation to allow application-specific dynamic weaving strategies, without introducing extra-overhead compared to static monolithic weavers.

6.5. Services management in large-scale environments

Participants: Mesaac Makpangou [correspondent], Ikram Chabbouh, Corina Ferdean, Nicolas Gibelin.

Web services technology standards enable description, publication, discovery of and binding to services distributed towards the Internet. However, current standards do not address the service selection issue : how did a consumer select the service, that matches its functional (e.g. operations' semantic) and non-functional (e.g. price, reputation, delay, response time) properties ? Most projects advocate automatic selection mechanism, advising adaptation or modification of the core facilities of web-services model (UDDI, WSDL, Client, Provider). These proposals also do not take advantage of distributed-systems' state of the art, mainly with respect to the collection and the dissemination of services' QoS.

First, we propose an extension of the initial model that permits automatic service selection, supports late binding, and collects and stores a number of metrics that characterize the quality of service [23], [18] . This proposal does impose any modification on UDDI registries nor on the provision of services. Precisely, the extension consists on a web-service access infrastructure, built on two basic mechanisms : web service cache proxy and web service QoS monitoring network. The proxies interacts with common UDDI registries to find suitable services for selection and to publish descriptions; and with QoS monitoring network to store/retrieve QoS metrics.

Secondly, we proposed an approach for improving the performance of a replicated service, while preserving the consistency of replicas needed for the service to run properly. This approach is conducted by two main goals: (1) maintaining the generality of the system with respect to the variety of services it supports (2) reducing the burden on user with respect to the service description that he has to provide

Precisely, we consider that the services have various demands on three main resources: the CPU, the disk I/O and the network bandwidth. The resource demands represent the service workload that the user is supposed to specify. The user has to specify the consistency options attached to the operations of the service.

We designed and implemented a generic replication framework, where each replication decision is resolved by an independent protocol, specialized according to the user demands. Precisely, the framework contains 6 protocols providing respectively: the replica consistency, the selection of the suitable replica (at connection-time and runtime re-selection), the processing of user alerts about bad perceived response time, the creation of new replicas, the migration of over-loaded replicas and the deletion of unuseful replicas.

The replica selection protocol is based on the abstract concept of metric estimator. In particular, we defined an estimator for the response time based on the resource demands of the service and the resource availabilities on hosts [21]. This estimator is defined as the sum of several components: the CPU service time, the disk I/O service time, the network transfer time, the CPU waiting time and the disk I/O waiting time. The former three components count for the time elapsed while the requests use respectively the CPU, the disk I/O and the network bandwidth resources. The latter two components count respectively for the time elapsed while the requests are the ready queue, respectively in the disk device queue. The main contribution of our work is the approximation of the waiting time by using the statistical regression technique. In this respect, estimating the waiting time for any workload relies on baseline measurements of the CPU waiting time (disk I/O waiting time) for some workloads under some CPU utilization (disk I/O utilization) points.

In a recent paper to appear in 2006 [22], we proposed a metrology infrastructure supporting the response time estimator, limited to only CPU and disk I/O resources. The replica hosts are aggregated into domains, containing nearby Autonomous Systems. There is a CPU monitor and a disk I/O monitor on each replica host. The measures of the hosts within a domain are transmitted pro-actively to a metrology server instance (attached to the domain). We are interested in quantifying the tradeoff between the response time estimation accuracy and the network bandwidth usage. We were able to find a threshold on the measures inaccuracy tolerated so as to maintain a reasonable accuracy of the response time. Currently, we complete this architecture with passive monitors collecting response time, piggybacked on requests propagated among the replicas.

6.6. Replication strategies, theorical aspects

Participants: Marc Shapiro [correspondent], Baptiste Canton, 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. Data replication has been studied extensively but separately in the context of databases, of distributed systems, and distributed algorithms. Consistency is almost completely ignored in the context of peer-to-peer systems, because serialising operations requires a consensus, which is inherently non-scalable. Taking semantics into account impacts consistency requirements, but different authors draw diverging conclusions.

Clearly, there is no single “right” approach to consistency, and many trade-offs are possible. However, a comprehensive approach to the problem has been missing, that considers all the aspects of consistency from first principles and enables a scientific study of the trade-offs. To this effect, we propose the Action-Constraint Formalism [25], [30]. It 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). This definition covers all known replication protocols, including optimistic or partial replication.

In the context of our research on consistency, we performed an extensive survey of optimistic replication algorithms. This was published in Computing Surveys [13]. We also continued previous work on reconciliation [11], and implemented the Action-Constraint Formalism in the Joyce framework [33].

We successfully applied the Rely-Guarantee approach to a non-trivial family of concurrent linked list algorithms. This work was written up in a paper, which is accepted for publication at Principles and Practice of Parallel Programming 2006 [32]. We also have encouraging preliminary results applying the technique to the proof of the Pastry ring protocol.

6.7. Dynamic replication in multi-agent systems

Participants: Olivier Marin [correspondent], Pierre Sens.

One software engineering technique for building scalable distributed software has emerged lately in the artificial intelligence research field, and appears to be both adequate and elegant: distributed agent systems. 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 characteristics that seem to emerge 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 agent software also proves to be adequate in the building of 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.

In addition, agents are defined as software elements which can equally be executed on any location, as long as the chosen host supports the required agent environment and provides adequate resources. This brings the multi-agent systems’ proneness to flexibility a step further. Distributing such systems over large scale networks should therefore tremendously increase their efficiency as well as their capacity, although it also brings forward the necessity of applying dependability protocols. Indeed, the more agents there are collaborating within an application, the higher the probability that at least one of them will fail eventually.

It is to be noted that most current multi-agent platforms and applications do not yet address, in a systematic way, the reliability issue. As mentioned earlier, multi-agent applications rely on the collaboration amongst agents. It follows that the failure of one of the involved agents can bring the whole computation to a dead end. Replicating every agent on different hosts may allow to easily bypass this problem. In practice, this is not feasible because replication is costly, and the multiplication of the agents involved in the computation can then lead to excessive overheads. Moreover, the criticality of a software element may change at some point of the application progress. Therefore dependability protocols ought to be optimally applied when and where they are most needed. In other words, only the specific agents which are temporarily identified as crucial to the application should be rendered fault-tolerant, and the scheme used for this purpose should be carefully selected. Replication is the one such type of scheme that is brought forward in the context of this work.

The reason for bringing forward the replication of data and/or computation is that it has been shown to be the only efficient way to achieve fault tolerance in distributed 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 critical applications, as well as applications with real-time constraints, both of 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. Active approaches should be chosen either if the failure rate becomes too high or if the application design specifies hard time constraints. In all other cases, passive approaches are preferable. In particular, active approaches must be avoided with non-deterministic software elements, for which a single input can lead to several different outputs, as the consistency between replicas cannot be guaranteed in this type of situation.

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) the possibility for applications to automatically choose which computational entities 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.

Currently, we are looking into means for software entities, be they agents or servers, to express their needs in terms of fault-tolerance: how critical they believe they are to the rest of the application, what kind of replication strategies may be applied to them, which resources will need to be present on replica-supporting nodes, ...

7. Other Grants and Activities

7.1. National initiatives

7.1.1. Grid Data Service - ACI MD (2003-2006)

Members: IRISA (Paris Team), ENS-Lyon (LIP - Remap Team), Regal

Objectives: The goal of this project is to propose an approach where the grid computation is decoupled from data management, by building a data sharing service adapted to the constraints of scientific grid computations. The main goal of this project is to specify, design, implement and evaluate a data sharing service for mutable data and integrate it into the DIET ASP environment developed by the ReMaP team of LIP. This service will be built using the generic JuxMem platform for peer-to-peer data management (currently under development within the PARIS team of IRISA, Rennes). The platform will be used to implement and compare multiple replication and data consistency strategies defined together by the PARIS team (IRISA) and by the REGAL team of LIP6.

7.1.2. Data Grid eXplorer (2003-2006)

Members: IMAG-ID, Laria, LRI, LAAS, LORIA, LIP Ens-Lyon, LIFL, INRIA Sophie Antipolis, LIP6, IBCP, CEA, IRISA INRIA Rocquencourt

Objectives: The goal of Data Grid Explorer is to build an emulation environment to study large scale configurations. Today, it is difficult to evaluate new models for data placement and caching, network content distribution, peer-to-peer systems, etc. Options include writing simulation environments from scratch, employing detailed packet-level simulation environments such as NS, local testing within a controlled cluster setting, or deploying live code across the Internet or a Testbed. Each approach has a number of limitations. Custom simulation environments typically simplify network and failure characteristics. Packet-level simulators add more realism but limit system scalability to a few hundred of simultaneous nodes. Cluster-based deployment adds another level of realism by allowing the evaluation of real code, but unfortunately the network is highly over-provisioned and uniform in its performance characteristics. Finally, live Internet and Testbed deployments provide the most realistic evaluation environment for wide-area distributed services. Unfortunately, there are significant challenges to deploying and evaluating real code running at a significant number of Internet sites. The main benefit of emulation is the ability to reproduce experimental conditions and results.

The project is structured horizontally into transverse working groups: Infrastructure, Emulation, Network, and Applications. The Regal team is leader for the Emulation working group.

7.1.3. Gedeon - ACI MD (2004-2007)

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

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.

7.1.4. Respire - ARA MDSA (2005-2008)

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

Objectives: RESPIRE project is financed by the ANR (ARA MDSA). It aims to apply techniques issued from the peer-to-peer communauty to share information distributed on large scale data bases.

7.2. European initiatives

7.2.1. Grant from Microsoft Research Cambridge

A Systematic Study of Data Replication Algorithms : PhD grant for Baptiste Canton.

8. Dissemination

8.1. Program committees and responsibilities

Luciana Arantes is:

- Member of the program committee of I WSO - Workshop de Sistemas Operacionais, SBC , 2005, Brazil.
- Member of the program committee of CDUR'2005 - *Journées Francophones sur la Cohérence des Données en Univers Réparti*, 2005, France
- Member of the "Commission de spécialiste" of the Paris 5 University.

Bertil Folliot is:

- Member of the program committee of The IASTED International Conference on Parallel and Distributed Computing and Networks, PDCN 2005, Innsbruck, Autriche, février 2005.
- Member of the program committee of the 2005 High Performance Computing and Simulation (HPC&S'05), Riga, Lettonie, juin 2005.
- Member of the program committee of the 4th International Symposium on Parallel and Distributed Computing, Lille, juillet 2005.
- Member of the program committee of the first International Workshop on System and Networking for Smart Objects (SaNSO2005), Fukuoka, Japon, juillet 2005.
- Member of the program committee of the 2005 International Conference on High Performance Computing and Communications, Sorrento, Italie, septembre 2005.
- Member of the program committee of the IFIP Symposium on Computer Architecture and High Performance Computing, Rio de Janeiro, Brésil, 2005, octobre 2005.
- Member of the "Commission de spécialistes" of the Paris 6 and Paris 9 Universities.

Mesaac Makpangou is:

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

Olivier Marin is:

- co-editor for IEEE distributed systems on-line in distributed agents topic.

Pierre Sens is:

- Member of the program committee of the 4th IEEE International Symposium on Network Computing and Applications (**IEEE NCA05**), 2005
- Member of the program committee of the 6th edition of the I2CS workshop (Internet Community Systems), Neuchâtel, Switzerland on June 26-28, 2006
- Member of the program committee of CFSE'2005- *Conférence française sur les systèmes d'Exploitation*. 2005, France.
- Member of the program committee of CDUR'2005 - *Journées Francophones sur la Cohérence des Données en Univers Réparti*, 2005, France
- Member of the "Commission de spécialistes" of Paris 6 and Lille 1 Universities.
- Elected member of the "Institut de Formation Doctorale" of Paris 6 University.
- Vice-chair of the LIP6 laboratory.

Marc Shapiro is:

- Founder and Chair of EuroSys, the European professional society on Computer Systems (European chapter of ACM SIGOPS).
- Member of the selection committee of Science Computer Programming, special issue on Memory Management.
- Member of the White Paper committee "Fostering Systems Research in Europe."
- Member of the Board of ASTI, Fédération des Associations Françaises des Sciences et Technologies de l'Information.
- Chair of the program commitee of Euro-Par 2005, topic Distributed Systems and Algorithms, Lisbon, Aug. 2005.
- Member, program committee of Fourth French Conference on Operating Systems (CFSE), Guérande, April 2005.
- Member of the program committee of CDUR 2005, Paris, Nov. 2005.
- Member of the program committee of the International Workshop on High-Performance Data Management in Grid Environments 2006, Rio de Janeiro, July 2006.
- Member of the program committee of the VLDB 2006 Workshop on Data Management in Grids, Seoul, Sept. 2006.

8.2. PhD reviews

Bertil Folliot was PhD rewiever of :

- Marc Segura Devillechaise, "Traitement par aspects des prblèmes d'évolution logicielle dans les caches Web", Université de Nantes, advisors : Pierre Cointe, Jean-Marc Menaud).
- Vivien Quenat, "Contribution à un exogiciel", Université de Grenoble, advisor : Jean-Bernard Stefani
- Julien Henriet, "Evaluation, optimisation et validation de protocoles distribués de gestion de la concurrence pour les interactions coopératives", Université de Franche-Comté, advisors : Eric Garcia, Jean-Christophe Lapayre

Pierre Sens was PhD rewiever of :

- Lamine Aouad, "Calcul numérique haute performance à grande échelle pair-à-pair", Université des Sciences et Technologies de Lille, advisor: Serge Petiton
- Raphael Chand, "Diffusion d'information à large échelle dans les systèmes de type publication /abonnement" Université de Nice-Sophia Antipolis, advisor: P. Felber
- Sylvain Dahan, "Recherche de services dans une fédération d'agents sur Grille", Université de Franche-Comté, advisors : Jean-Marc Nicod, Laurent Philippe .
- Arnas Kupsys, "JMSGroups: JMS Interface for Group Communication", EPFL , advisor: André Schiper.
- Jean-Michel Nlong II, "Conception et réalisation d'un intergiciel pour la résilience d'applications parallèles distribuées sur un intranet et Internet", Institut National Polytechnique de Grenoble, advisors: B. Plateau, M. Tchuente
- Olivier Soyez, "Stockage dans les systèmes pair à pair", Université de Picardie Jules Verne d'Amiens, advisor: V. Villain

Marc Shapiro was PhD reviewver of :

- Gérald Oster, "Réplication optimiste et cohérence des données dans les environnements collaboratifs répartis," LORIA, advisors: Claude Godard and Pascal Molli.
- Cécile Lepape, "Contrôle de Qualité des Données Répliquées sur un Cluster", Université Paris 6, Advisors: Stéphane Gançarski and Patrick Valduriez.

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
  - Distributed systems, DESS IRS, Université de Versailles Saint-Quentin en Yvelines
  - Performance of distributed systems, DEA MISI, Université Versailles Saint-Quentin
  - Large Scale Data sharing, DEA IFA, Université de Marne-La-Vallee
  - Distributed systems and applications, Institut Superieur de Technologies et de management (ISTM)
  - Systems and networks, Master, Pôle Universitaire Leonard de Vinci
- Oliver Marin
  - Operating system programming, Master d'Informatique, Univeristé Paris 6
  - Operating system Principles, Licence d'Informatique, Univeristé Paris 6 Université Paris 6
  - Parallel and distributed systems,Master d'Informatique, Université Paris 6
  - Client/server arcitecture in 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

9. Bibliography

Major publications by the team in recent years

[1] L. B. 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, 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.

[4] 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.

[5] 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.

[6] 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.

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

Doctoral dissertations and Habilitation theses

[8] G. THOMAS. *Applications actives - Construction dynamique d’environnements d’exécution flexibles homogènes*, Ph. D. Thesis, May 2005.

Articles in refereed journals and book chapters

[9] G. ANTONIU, L. BOUGÉ, E. CARON, F. DESPREZ, M. JAN, S. MONNET, P. SENS. *Future Generation Grids*, CoreGrid Series, To appear, vol. 2, chap. GDS: An Architecture Proposal for a Grid Data-Sharing Service, Springer Verlag, 2005.

[10] M. BERTIER, L. ARANTES, P. SENS. *Distributed Mutual Exclusion Algorithms for Grid Applications: a Hierarchical Approach*, in "Journal of Parallel and Distributed Computing (JPDC)", To appear, 2005.

[11] Y. HAMADI, M. SHAPIRO. *Pushing log-based reconciliation*, in "Int. J. on Artif. Intelligence Tools (IJAIT)", vol. 14, n^o 3–4, jun 2005, p. 445–458, http://www.research.microsoft.com/~youssefh/Papers/ijait05Log.pdf.

[12] F. OGEL, G. THOMAS, B. FOLLIOT, I. PIUMARTA. *Concurrent Information Processing and Computing*, D.

GRIGORAS, A. NICOLAU (editors). , Nato Science Series III, vol. 195, chap. Application-Level Concurrency Management, IOS Press, 2005, p. 19–30.

[13] Y. SAITO, M. SHAPIRO. *Optimistic Replication*, in "Computing Surveys", vol. 37, n^{o} 1, March 2005, p. 42–81, http://doi.acm.org/10.1145/1057977.1057980.

[14] G. THOMAS, F. OGEL, A. GALLAND, B. FOLLIOT, I. PIUMARTA. *Building a Flexible Java Runtime upon a Flexible Compiler*, in "Special issue System and networking for SmartObjects of IASTED International Journal of Computers and Applications", vol. 27, n^{o} 1, 2005, p. 27–34.

Publications in Conferences and Workshops

[15] S. AKNINE, O. MARIN. *Role of Replication Planning for Fault Tolerant Multiagent Systems*, in "Fifth European Workshop on Adaptive Agents and Multi-Agent Systems, Paris, France", March 21-22 2005, p. 221–228.

[16] M. BERTIER, L. ARANTES, P. SENS. *Algortihme d'exclusion mutuuelle pour les grilles : une approche hiérarchiqu e*, in "Actes de la 4ème Conférence Française sur les Systèmes d'Exploitation (CFSE'4), Le Croisic, France", May 2005.

[17] 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.

[18] I. CHABBOUH, M. MAKPANGOU. *A Configuration Tool for Caching Dynamic Pages*, in "9th International Workshop WCW, Beijing, China", Springer, October 2005.

[19] C. CLÉMENT, B. FOLLIOT. *Ordonnanceurs hiérarchiques adaptables*, in "Actes de la Manifestation des Jeunes Chercheurs francophones dans les domaines des STIC (Majestic), Rennes, France", November 2005.

[20] C. FERDEAN, M. MAKPANGOU. *A Fine-Grained Customizable Consistency Protocol for Replicated Data Objects*, in "Journées Francophones sur la Cohérence des Données en Univers Réparti (CDUR), CNAM, Paris, France", November 2005.

[21] C. FERDEAN, M. MAKPANGOU. *Exploiting Application Workload Characteristics To Accurately Estimate Replica Server Response Time*, in "International Symposium on Distributed Objects and Applications (DOA), Agia Napa, Cyprus", October 2005.

[22] C. FERDEAN, M. MAKPANGOU. *A Response Time-Driven Server Selection Substrate for Application Replica Hosting Systems*, in "International Symposium on Applications and the Internet (SAINT'06)", To appear, 2006.

[23] N. GIBELIN, M. MAKPANGOU. *Efficient and transparent Web-Services selection*, in "ICSOC05 : International Conference on service oriented computing", B. BENATALLAH, F. CASATI, P. TRAVERSO (editors). , vol. 3826, Springer-Verlag GmbH, decembre 2005, p. 527 - 532.

[24] A. HACHICHI, G. THOMAS, C. MARTIN, S. PATARIN, B. FOLLIOT. *A Generic Language for Dynamic*

Adaptation, in "European Conference on Parallel Processing, EuroPar 2005, Lisboa, Portugal", Lecture Notes in Computer Science 3648, Springer-Verlag, August 2005, p. 40–49.

[25] 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.

[26] F. OGEL, G. THOMAS, B. FOLLIOT. *Supporting Efficient Dynamic Aspects through Reflection and Dynamic Compilation*, in "20th Annual ACM Symposium on Applied Computing, Santa Fé, New Mexico", March 2005, p. 1351–1356.

[27] S. PATARIN, M. MAKPANGOU. *Pandora: An Efficient Platform for the Construction of Autonomic Applications.*, in "Self-star Properties in Complex Information Systems", Lecture Notes in Computer Science, vol. 3460, Springer, 2005, p. 291-306.

[28] F. PICCONI, P. SENS. *Using incentives to increase availability in a DHT*, in "Third International Workshop on Hot Topics in Peer-to-Peer Systems (in conjunction with IEEE IPDPS), Rhodes Island, Greece", To appear, IEEE press, April 2006.

[29] N. PREGUIÇA, C. BAQUERO, J. LEGATHEAUX MARTINS, M. SHAPIRO, P. S. ALMEIDA, H. DOMINGOS, V. FONTE, S. DUARTE. *FEW: File Management for Portable Devices*, in "Proc. Int. W. on Software Support for Portable Storage, San Francisco, CA, USA", March 2005, http://asc.di.fct.unl.pt/few/papers/few-iwssps2005.pdf.

[30] M. SHAPIRO, N. KRISHNA. *The three dimensions of data consistency*, in "Journées Francophones sur la Cohérence des Données en Univers Réparti (CDUR), CNAM, Paris, France", November 2005, http://www-sor.inria.fr/~shapiro/papers/cdur2005.pdf.

[31] J. SOPENA, L. ARANTES, M. BERTIER, P. SENS. *A fault-tolerant token -based mutual exclusion algorithm using a dynamic tre e*, in "Euro-Par'05 - Parallel Processing, Lisboa, Portugal", Lecture Notes in Computer Science 3648, Springer-Verlag, August 2005.

[32] V. VAFEIADIS, M. HERLIHY, T. HOARE, M. SHAPIRO. *Proving Correctness of Highly-Concurrent Linearisable Objects*, in "Principles and Practice of Parallel Programming (PPoPP), New York, USA", Preliminary version: INRIA RR-5716, March 2006.

Internal Reports

[33] J. O'BRIEN, M. SHAPIRO. *An Application Framework for Collaborative, Nomadic Applications*, Rapport de recherche, n^o RR-5745, inria, rocquencourt, November 2005, http://www.inria.fr/rrrt/rr-5745.html.

[34] V. VAFEIADIS, M. HERLIHY, T. HOARE, M. SHAPIRO. *Proving Correctness of Highly-Concurrent Linearisable Objects*, Rapport de recherche, n^o RR-5716, Institut National de la Recherche en Informatique et Automatique (INRIA), Rocquencourt, France, October 2005, http://www.inria.fr/rrrt/rr-5716.html.

Miscellaneous

[35] O. MARIN. *Adapting fault tolerance in highly dynamic environmentsMonterey Workshop, Laguna Beach, CA, USA*, September 22-24 2005.

[36] M. SHAPIRO, Y. BERBERS, W. ZWAENEPOEL, T. HARRIS. *Systems research, education and industry in Europe*, June 2005, http://www-sor.inria.fr/~shapiro/papers/Systems_research,_education_and_industry_in_Europe.pdf, Text submitted to the European Commission.