| | | | |
|---|---|---|---|
| **Acronyme** | ConcoRDanT | | |
| **Titre du projet** (en français) | Les CRDT pour la cohérence sans contrôle de concurrence dans les nuages et les systèmes pair-à-pair | | |
| **Titre du projet** (en anglais) | CRDTs for consistency without concurrency control in Cloud and Peer-to-Peer systems | | |
| **Comité d'Évaluation référence (CE)**[1] | SIMI2 | | |
| **Projet multidisplinaire** | ☐OUI ☒NON<br>Si oui, indiquer l'intitulé du second CE : SIMI3 | | |
| **Coopération internationale** (si applicable) | Le projet propose une coopération internationale<br>☐ avec les États-Unis (accord ANR/NSF)<br>☒ autres pays | | |
| **Aide totale demandée** | 321 742 € | **Durée du projet** | 36 mois |

# SOMMAIRE

# ConcoRDanT: CRDTs for consistency without concurrency control in Cloud and Peer-To-Peer systems

**Abstract**

Massive computing systems and their applications suffer from a fundamental tension between scalability and data consistency. Avoiding the synchronisation bottleneck requires highly skilled programmers, makes applications complex and brittle, and is error-prone.
The ConcoRDanT project investigates a promising new approach that is simple, scales indefinitely, and provably ensures eventual consistency. A Commutative Replicated Data Type (CRDT) is a data type where all concurrent operations commute. If all replicas execute all operations, they converge; no complex concurrency control is required. We have shown in the past that CRDTs can replace existing techniques in a number of tasks where distributed users can update concurrently, such as co-operative editing, wikis, and version control. However CRDTs are not a universal solution and raise their own issues (e.g., growth of meta-data).
The ConcoRDanT project engages in a systematic and principled study of CRDTs, to discover their power and limitations, both theoretical and practical. Its outcome will be a body of knowledge about CRDTs and a library of CRDT designs, and applications using them. We are hopeful that significant distributed applications can be designed using CRDTs, a radical simplification of software, elegantly reconciling scalability and consistency.

## 1 Contexte et positionnement du projet / Context and positioning of the proposal

Massively distributed computing infrastructures are becoming central to our economies and to our lifestyles. Cloud or Peer-to-Peer systems permit economies of scale through resource sharing, adaptation to the users' changing needs, a seemingly infinite supply of computing resources, outsourcing of management, etc.

Clients require fast, always-on access and dependability; providers need to scale the infrastructure without obstacles. One key to such high availability, performance and durability is replicating data at many locations. This works very well for a large class of "embarrassingly parallel" applications, where the data is either read-only, single-writer, or accessed in disjoint chunks. However, when data is shared and *mutable* (i.e., modifiable), updating one replica may cause the other to be inconsistent. Many applications cannot tolerate inconsistency, at least not without great complexity. Yet there is a *fundamental tension* between consistency, which requires synchronisation between remote processes, and scalability, which requires independence.

The ConcoRDanT proposal addresses this tension, thanks to a novel concept, the *Commutative Replicated Data Type* or CRDT. The insight is that, if concurrent operations commute, their execution order doesn't matter.[2] Then, if every replica receives every operation, they converge automatically, without concurrency control. Replicas can be close or far, or even disconnected for long durations. Commutativity comes naturally between independent pieces of data; our challenge is to design *useful, efficient, general-purpose shared data types whose operations commute when concurrent.*

[2] With respect to some data type, we distinguish two subsets of operations. The first one is composed only of operations that all commute with one another when concurrent. The second is the complementary to the first. Abusing mathematical rigour somewhat, we will call the former the "commuting operations" and the latter the "non-commuting operations."

The advantages of commutativity in parallel computing are well known. However, the issue of *designing* shared data types for commutativity has been so far neglected. The ConcoRDanT participants have recently made substantial contributions to the state of the art, designing non-trivial CRDTs that provably converge and maintain useful invariants. They used these CRDTs in practical applications. They showed that they have excellent performance and that they scale to massive distributed systems. This required solving some interesting difficulties. For instance we carefully analysed the application to minimise its requirements, i.e., we showed that invariants used in a classical design were too strong. We had to design a dense set of identifiers that are at the same time unique, ordered and compact. Two important issues were avoiding the growth of meta-data overhead, and managing the co-existence of both commutative and non-commutative operations. We suspect that such difficulties are typical of CRDTs in general.

The ConcoRDanT project investigates the promising CRDT approach, which is simple, scales indefinitely, and provably ensures eventual consistency.[3] We have shown in the past that CRDTs can replace existing techniques in a number of tasks where distributed users can update concurrently, such as co-operative editing, wikis, and version control. However CRDTs are not a universal solution and raise difficult issues (e.g., growth of meta-data).

The ConcoRDanT project engages in a systematic and principled study of CRDTs, to discover their power and limitations, both theoretical and practical. The major outcome of ConcoRDanT shall be a body of knowledge about CRDTs, a library of CRDT designs, and applications using them.

We are reasonably hopeful that we can build realistic, interesting and useful distributed applications by carefully combining CRDTs from this library. If this happens, it will constitute a significant breakthrough, reconciling scalability and consistency in an elegant, simple and principled way. Given the very preliminary state of the art, this most favourable result is not certain; however, even without it, ConcoRDanT will advance the state of the art thanks to its systematic exploration of the CRDT design space.

The ConcoRDanT project brings together the world experts on CRDTs. It builds upon our previous research on large-scale replication and consistency, some of it joint work between the participants. Thanks to this experience, we are well aware of the possibilities, and also of the difficulties, limitations and risks.

Several existing and past projects were interested on consistency of replicated data for distributed system, but none address directly the topic of operation commutativity. The INRIA ARC Recall (2006-2007) was the first project to target consistency on peer-to-peer network for collaborative editing. One of the result of this project was the Woot algorithm, a first kind of CRDT. The RNTL XWiki Concerto (2006-2009) allowed the industrial transfer of Woot into XWiki platform. The ARA Respire (2006-2008) also focused on consistency in peer-to-peer networks but using the Action-Constraint-Framework (ACF) that only expresses non-commutative operations as one of the constraints to ensure. It does not aim to relax invariants of non-commutative operations to make them commutative in contrast with the objectives of the ConcoRDanT project. The ANR DataRing (2009-2012) project addresses the more general problem of peer-to-peer data sharing for online communities, including data search, privacy, cache management and semantic. It considers data replication as one of the offered services. Such service could be provided by means of CRDTs. The objective of the Wiki 3.0 project (2010-2011) is the development of a

[3] In the context of a replicated shared datum, *eventual consistency* means that, as users update their replicas they may diverge, but, if users stop making updates, all replicas eventually converge to the same correct value that includes all updates [29, 35].

wiki platform that addresses the three major evolution axes of collaborative Web: real-time collaboration, social interaction integrated into the production (chat, micro-blogging, etc.) and on-demand scalability (cloud computing). One of the issues investigated by this project is the suitability of existing CRDT algorithms for real-time editing.

# 2 Description scientifique et technique / Scientific and technical description

## 2.1 État de l'art / Background, state of the art

### 2.1.1 Scalability and consistency

Today's computing systems must scale to enormous numbers of computers, and to worldwide distribution, with high and variable communication latencies. This constitutes an overarching requirement of both Cloud [8, 14, 41] and Peer-to-Peer systems [11, 34, 36]. Much research has targeted scalability of communication (e.g., event delivery and multicast protocols), assuming either single-writer data (Web), read-only data (e.g., multimedia content), or data-parallel updates (e.g., MapReduce). These are the easy cases.

However, operators and users of massive-scale computing systems are increasingly concerned with managing updates [1, 13]. They realise that the above techniques are not sufficient when dealing with shared mutable data, because of the consistency requirement. Even in applications that ostensibly do not demand consistency (e.g., MapReduce computations), the system itself requires consistency of configuration metadata describing the computation [8]. Both academia and industry are actively searching for efficient synchronisation mechanisms that provide well-defined guarantees; see for instance recently HP's Sinfonia [1] or Google's Chubby [9],

Synchronisation creates a scalability bottleneck, not only because it slows things down, but also because it causes dangerous feedback oscillations in the infrastructure [8]. Avoiding this bottleneck, developers work hard to program around synchronisation, using *optimistic, eventual consistency* techniques [35, 41]. For instance, the Amazon Shopping Cart is designed for availability over consistency; it uses optimistic techniques based on commutative operations (i.e., set union) [13]. Quoting a report on the LADIS 2008 workshop on large-scale computing [8]:

> Thus for our keynote speakers [four principal architects from IBM Websphere, Microsoft Cloud Computing Initiative, and eBay], "fear of synchronization" was an overarching consideration that in their eyes, mattered far more than the theoretical peak performance of such-and-such [a protocol]. [...] The bigger –the overarching– challenge is to find ways of transforming services that might seem to need locking into versions that are loosely coupled and can operate correctly without locking — to get [synchronisation] off the critical path [...], moving towards a decentralised convergence behaviour in which server nodes are (as much as possible) maintained in loosely consistent but transiently divergent states, from which they will converge back towards a consistent state over time.

However, this is complex at best and requires extremely highly-skilled programmers. The current manual, best-effort approach provides no guarantees and is prone to error. For instance, the Amazon S3 service was recently disabled for an entire day because of a single corrupted value, forcing the whole of Amazon to shut down and restart its operation [3].

Previous research on eventual consistency includes the theory of causal consistency [2], systems such as Bayou [29, 39], and more recently the Telex middleware from partner INRIA Regal [6].

### 2.1.2 Commutativity and shared data

It is well known that commutative operations are advantageous in parallel computing. A number of papers study the advantages of commutativity for concurrency and consistency control [4, 7, 42, for instance]. Systems such as Psync [24], Generalised Paxos [20], Generic Broadcast [28], IceCube [31] and Telex [6] make use of commutativity information to relax consistency or scheduling requirements. If all concurrent operations commute, then the simple "causal consistency" approach is sufficient to ensure eventual consistency [2, 32]. However, none of the above works addresses the issue of designing shared data structures supporting commutativity.

Weihl studied commutativity-based concurrency control for abstract data types [42]. He distinguishes between forward and backward commutativity. They differ only when operations fail their pre-condition. Our approach considers only operations that succeed at the submission site, and we ensure by design that they won't fail at replay sites.

The so-called Last-Writer-Wins (LWW) approach is widely used in databases and file systems [18, 35]. It considers a write operation that also sets a hidden "timestamp" attribute. If users concurrently assign different replicas, the one with the highest timestamp overwrites the other. Technically, this satisfies commutativity. Note however that the write with the lower timestamp is not durable (it is lost); it follows that LWW cannot ensure any useful guarantees. In contrast, CRDTs should ensure *genuine* commutativity, i.e., post-conditions of operations should be durable.

Roh et al. [33] independently proposed a concept similar to CRDTs. To make concurrent assignments commute, they propose a precedence order, similar to LWW. Roh does not consider the case where concurrent updates should be merged, not lost, as in co-operative editing.

Operational transformation (OT) [16, 22, 37, 38] studies collaborative editing. OT attempts to transform non-commuting operations to make them commute after the fact. In contrast, the CRDT approach is to design operations to commute in the first place. OT transforms the operations to be executed to take into account the effects of concurrent operations. Many OT algorithms require substantial meta-data or a centralised server to detect concurrent operations, which do not scale in dynamic cloud and peer-to-peer environments. Others, such as Jupiter [25], used by Google Wave, require a unique ordering of operations, which is not suitable as it increases network delays. CRDT algorithms have much weaker requirements, which makes them efficient in environments subject to churn and failures.

Baquero [5] studies convergence conditions for replicated objects and states that their operations should be idempotent, commutative and associative. Thus CRDTs can be derived from abelian groups, such as an integer register with "add" and "subtract" operations, or a set with "insert-element" as its only operation. However, it has unusual semantics, since concurrently-inserted elements must be distinct. This suffices, for instance, to implement a mailbox. However, it is not practical, as data grows without bound.

Similarly WOOT, a CRDT for concurrent editing [27] has a large metadata overhead and grows without bound.

### 2.1.3 Our progress so far

Little is known about non-trivial CRDTs [5, 22, 33], i.e., ones that support complex structures useful for high-level computation, remain of reasonable size, and are durable. However, the ConcoRDanT proposers have made important advances in this direction. The Treedoc [21, 30] and Logoot [43] data structures were designed by the proposers. Both provide the abstraction of a replicated sequence, and were designed for concurrent editing applications. These designs have been proved correct and are efficient in storage, computation and communication costs. They have been validated, with very promising performance measurements, using traces from existing massive-scale systems such as Wikipedia or SVN.

Let us consider Treedoc in more detail. It maintains a sequence of elements and supports *insert-at-position* and *delete-at-position* operations. An element is identified by a unique identifier: the design challenge to keep identifiers short and to minimise overhead. Identifiers must be globally unique, stable, and ordered identically to the sequence. Furthermore, it must always be possible to create a new identifier between two existing ones.

Internally, we use an extended binary tree for identification. There are two main sources of overhead: accumulation of deleted elements ("tombstones"), and tree unbalance. Therefore, we propose a garbage collection mechanism that removes tombstones and rebalances the tree. A compacted Treedoc reduces to a sequential array, with zero overhead. Compaction requires a consensus and does not commute with inserts and deletes, but this is not a problem, because insert/delete operations take precedence and a concurrent compaction aborts.

To ensure that compaction scales, we propose a flexible two-tier architecture: A small, stable *core* supports both updates and consensus. It coexists with a unlimited, uncontrolled, dynamic *nebula* supporting only updates. A novel protocol allows a nebula site to catch up with the core's state.

We validated this design with a benchmarking study, based on traces from existing editing histories.

Its design presents original solutions to scalability issues, namely restructuring the tree without violating commutativity, supporting very large and variable numbers of writable replicas, supporting disconnected nodes, and leveraging the data structure to ensure causal ordering without vector clocks.

To overcome the challenges of practicality and scalability, we explored some innovative solutions. Each element has a unique, system-wide, compact identifier that does not change between garbage collections. Garbage collection is a requirement in practice; it is disruptive and requires consensus, but it has lower precedence that updates, and it is not in the critical path of applications. We side-step the non-scalability of consensus by dividing sites into two tiers with different roles.

Another CRDT, the Multilog, designed by Regal, provides the abstraction of a directed graph, and forms the basis of the Telex middleware for co-operative applications [6]. Its operations are to create vertices and edges between them. Telex ensures that the graph conforms to some specific structural constraints and grows in a well-defined way. This allows efficient garbage collection and avoids indefinite growth.

This work allowed us to identify some general properties for the design of CRDTs [21]. The goal of this proposal is to generalise this approach to manage data in massive-scale environments, by a systematic and principled study of CRDTs.

## 2.2 Objectifs et caractère ambitieux, novateur du projet / Rationale highlighting the originality and novelty of the proposal

**Technical objectives** Future computer systems will hold massive numbers of objects, replicated and shared by numerous users, widely distributed over the network. The tension between consistency and scalability is a major roadblock to the development of massive decentralised applications that share mutable data, such as databases, collaboration environments, or distributed games. Our larger scientific agenda is to study consistency of decentralised, uncoordinated updates in this environment, a very difficult problem.

CRDTs represent a *radically new approach*. A CRDT suffers no conflicts, hence, have no need for costly concurrency control. CRDTs are not a universal solution, but, perhaps surprisingly, we were able to design highly useful CRDTs. CRDTs are appealing because they are easy to understand and to use. This research direction is promising as CRDTs provide eventual consistency in the large scale at a low cost.

So far we have successfully designed CRDTs for specific purposes, and we have an intuition of how to design CRDTs and what are their inherent limitations. In the context of this proposal, we will investigate the generalisation of CRDT or CRDT-like techniques for managing replicated data in massive-scale, decentralised computing environments and their applications. More specifically, the objectives of ConcoRDanT are: To survey the state of the art and requirements of existing CRDT (or CRDT-like) designs, and of industrial practice (Task 2);[4] To establish formally what is theoretically possible and not possible, and how CRDTs relate to known synchronisation classes (Task 3); To generate a comprehensive collection of widely-useful CRDT designs and techniques (Task 4); To investigate the coexistence of commutativity with non-commutativity, and to extend the CRDT approach to stronger invariants (Task 5); and To implement an open-source CRDT library, to use our CRDTs in applications, and to evaluate their performance on real traces or realistic benchmarks (Task 6).

**Challenges** Since little is known about CRDTs, and because of the inherent difficulties of large-scale distributed computing, success is not guaranteed. Despite our early successes, and our ideas for future CRDTs, the area may turn out to be less rich than anticipated.

Our preliminary experience shows a number of issues that need to be solved. The asynchronous nature of CRDTs causes meta-data to accumulate, requiring garbage collection, a global operation. Although the class of properties that can be maintained using only commutative operations is not known precisely (this is the charter of Task 3), it is already clear that they are relatively weak, and that occasional forays into non-commutativity will be inevitable in practice (to be studied in Task 5).

On the other hand, results so far are very promising, and we have been able to find original solutions for the difficulties. It is precisely this challenge that makes the project exciting. We know already that the large operators such as IBM, eBay, Google, Facebook, SAP, Amadeus, etc., are devoting very substantial amounts of person-years of engineering to address, in an *ad hoc* way, the avoidance of synchronisation for scalability, at the expense of consistency and correctness. Thus, even if CRDTs turn out to be a disappointment, our systematic, scientific, principled study of the design space will represent a significant advancement of the state of the art.

[4] Task 1 is project co-ordination; this numbering is imposed by the ANR submission system.

**Originality** The advantages of commutativity are well-known, but how to achieve commutativity has not been well studied. Although CRDTs designs have occasionally been published, we are the first (to our knowledge) to identify the concept, to address the design of CRDTs, and to engage in a systematic study. So far, only a handful of CRDTs are known. Furthermore, so far, the study of CRDTs has been *ad hoc*, on a case-by-case basis. The ConcoRDanT project proposes a *systematic* and *principled* study of the CRDT concept, from both a fundamental and a practical perspective.

**Anticipated scientific and technical results** The scientific result of the project will be a number of publications both in the area of distributed systems and algorithms, and in that of cooperative work. The technical result of the project will be fundamental and practical knowledge, including libraries of CRDTs that we will make available in open source. If successful, this will enable a *major simplification* in the way massive-scale distributed systems are built, used and managed.

**Success criteria** As scientists, our primary measure of success is publication. In the longer term, a more interesting measure of success is the take-up of results, by industry and by the open source community. Such widespread adoption may require an engineering effort that goes beyond the scope of this project. Within the context of the scientific and technical objectives of the project, the metric for success is the influence of our studies within the research community, and both the academic and industrial take-up of our algorithms.

# 3 Programme scientifique et technique, organisation du projet / Scientific and technical programme, project management

## 3.1 Programme scientifique et structuration du projet / Scientific programme, specific aims of the proposal

The goal of this project is a systematic exploration of the CRDT concept and design space, in order to provide both knowledge and code. The successful development and use of CRDTs has the potential to substantial simplify the design and implementation of massive-scale distributed systems and applications that share mutable data. Our research programme is informed by our knowledge of distributed computing.

Our approach will involve theoretical studies, algorithm design, and validation through formal verification, simulation studies, performance evaluation, and actual implementation and measurement. This research will be informed by advances in the state of the art and by the actual requirements of applications in massive computing environments.

The project is divided into six tasks. The division is motivated by our preliminary experience with CRDT design and our feel of the challenges.

Coordination and management constitute Task 1 (numbering imposed by the ANR submission system). It obviously lasts the whole duration of the project.

Task 2 takes place during the first year, representing the initial work of the PhD students. It helps prepare the work of the other technical tasks. Its outcome will be a comprehensive survey of the state of the art and requirements for Tasks 3 and 4. It will survey the existing theory

of commutativity as it relates to the goals of the project. It will also survey existing CRDT (or CRDT-like) designs, and industrial practice such as Amazon Shopping Cart, Google Wave, Wikis, etc.

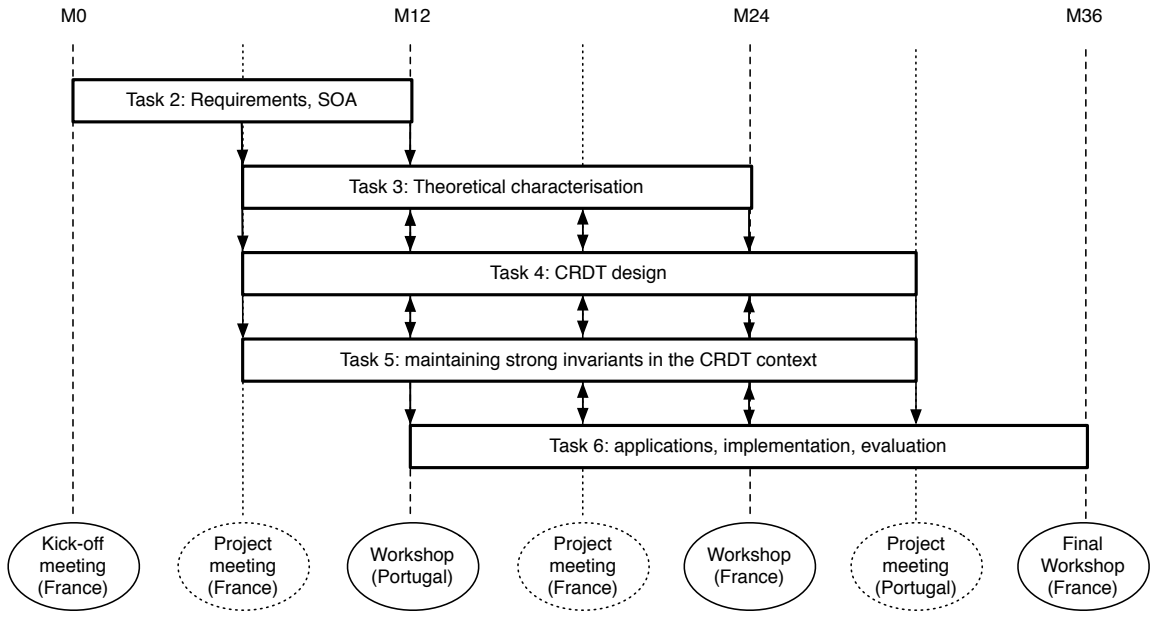
Task 3 is the theoretical part of the project. It runs from Month 6, for 18 months, under the responsibility of the post-doc. It aims to characterise formally what is possible and not possible, e.g., in terms of classes of invariants or post-conditions that can be maintained.

Task 4 aims to generate a comprehensive collection of useful CRDT designs. It takes into consideration the state of the art and requirements from Task 2, theoretical understanding from Task 3, and suggestions from Task 5, In turn its results shall inform the other tasks. Accordingly, it runs from M6 to the end.

Since pure CRDTs are not universal, Task 5 investigates the coexistence between commutative and non-commutative operations, extending the CRDT approach to support strong and high-level invariants. It interacts closely with the other technical tasks, and runs from M6 to the end.

Finally, Task 6 is chartered with implementing an open-source library of CRDTs, implementing practical applications that use them, and evaluating their performance on real traces and realistic benchmarks. It takes inspiration from Tasks 2, 3, 4 and 5, and its findings feed back into the these tasks.

The following diagram illustrates the relationships between the technical tasks of the project.




## 3.2 Coordination du projet / Project management

### Task 1: Project co-ordination

| **Regal (leader)** | Score | UNL | M1 (start) |
|---|---|---|---|
| 6 p×m | 2.2 p×m | 1 p×m | M36 (end) |

There are two main management activities: planning and coordinating partners. Management aims to maintain a highly reactive atmosphere, to achieve strong interaction and coordination

between partners, to progress according to the planned milestones and deliverables, and to achieve the project objectives with a good level of quality.

Project coordination is the responsibility of the project leader, Marc Shapiro of Regal. The leader organises meetings and reviews, monitors progress, keeps track of deliverables, prepares the project reports, and manages the budget. Given the research focus of the project and the small number of partners, the management style can remain lightweight, under direct supervision of the project leader, coordinating with the task leaders.

There will be either two PhD students working half-time on the project, one at Regal and one at Score, performing extended visits to the other groups, or, if possible, a single student co-advised by Regal and Score. Either way, this will help ensure strong partner co-ordination, integration of work, and good exchange of information.

The three groups have a history of informal collaboration; indeed, they have authored several joint papers. To facilitate the coordination of the project, Nuno Preguiça and his students will visit the two French groups for at least two months each summer, with the possibility of other shorter exchanges as needed.

The primary background of Regal and UNL is distributed systems and algorithms; they will be in charge of selecting publication venues in that area, e.g., EuroSys, PODC or ICDCS. The focus of Score is collaborative work, and they will select publication venues accordingly, e.g., CSCW, GROUP or CollaborateCom.

A two-day kickoff meeting will start off the project. An open workshop will be organised in France at the end of the project, to demonstrate and disseminate its findings. We will organise a partner meeting and a workshop every year, attended by all the people working on the project. Two-thirds of the meetings will be in France, the others in Portugal. A telephone conference will take place at bi-monthly intervals.

| **Deliverables** | **Deadlines** |
|---|---|
| D1.1: Kick-off meeting | M0 |
| D1.2: Website | M3 + continuous updates |
| D1.3: Yearly workshop | M12 |
| D1.4: Yearly workshop | M24 |
| D1.5: Final open workshop | M36 |
| D1.6: Final report | M36 |
| Coordination meetings | M6, M18, M24 |

### 3.3 Description des travaux par tâche / Detailed description of the work organised by task

**Task 2: Requirements for CRDTs and state of the art**

| **UNL (leader)** | Regal | Score | M1 (start) |
|---|---|---|---|
| 10 p×m | 8 p×m | 5 p×m | M12 (end) |

**Objectives** The objective of this task is a survey of the state of the art and of requirements related to CRDTs. We will study existing replicated data types, including ones that are not ostensibly CRDTs, but have similar goals. We will also examine what data types might be re-engineered to decrease dependence on non-commutativity (called quasi-CRDTs). Finally, we will analyse massively distributed applications used in industry, in order to derive requirements for maintaining consistency in these applications.

**State of the art** We refer to Section 2.1.2 for a comprehensive state of the art.

**Chosen methods and anticipated solutions** Classically, we will get our information from a literature survey, from the web, and from informal exchanges at conferences. We will draw on our own previous experience in designing distributed systems and applications and will study open-source collaboration systems. We will also make use of our good industrial contacts at Google, Facebook, SAP, IBM, and with French SMEs that are deploying social networks.

We aim to identify different techniques used for managing replicated data, and scenarios in which CRDTs might can be used, in the context of cloud and P2P computing. The following scenarios seem to be good candidates. First, collaborative editing among distributed users, a popular application scenario supported by Wikipedia, Google Docs, Microsoft Groove, and Google Wave for instance. Second, file systems, including version control systems, or HDFS (used in the Hadoop cloud computing infrastructure). Third, key-value maps, such as the ones used in DHTs and in Amazon Dynamo for maintaining data.

**Work plan** We will perform a literature survey, searching for CRDTs or similar approaches, and will perform an analysis of existing systems. We will analyse requirements of existing systems (e.g., wikis) and compare the known CRDTs with these requirements.

We will analyse centralised and decentralised collaborative version control systems, including as CVS, Subversion, Git, Mercurial and Darcs. Accordingly, CRDT requirements for file systems exporting file and directory create, delete, move, and modify operations will be studied. Concurrency control approaches used in various distributed file systems will be investigated, and their advantages and limitations will be compared with CRDTs.

Of particular interest is Google Wave, and the requirements of building a CRDT for tree-structured XML documents, on which Wave is based.

We will investigate the CRDT in the context of data management issues in cloud computing. For instance, we may analyse Amazon Dynamo, with particular focus on the requirements of its key-value map data type.

In each case, we aim to to determine whether using CRDTs, or CRDT-inspired techniques for achieving commutativity, can satisfy the requirements.

| **Deliverables** | **Deadlines** |
|---|---|
| D2.1: Literature survey | M6 |
| D2.2: Survey of relevant systems | M12 |

**Success criteria** The outcome of this task shall be a taxonomy of requirements, techniques, challenges, limitations, etc., related to CRDTs. If sufficiently clear and comprehensive, it will ease the other tasks. In the best case, this task might lead to discovering new data types and new application domains where CRDTs could be used. We aim to publish these results in significant venues.

**Risks** Our survey of massive distributed applications could disappoint, if, for instance, we find too few scenarios where CRDTs are useful.

## Task 3: Theoretical characterisation of CRDTs

| **Regal (leader)** | Score | UNL | M6 (start) |
|---|---|---|---|
| 12 p×m | 5 p×m | 5 p×m | M24 (end) |

**Objectives** This task aims to characterise the scalability and the class of properties that can be achieved in a CRDT. We will investigate how CRDTs can be defined formally.

**State of the art** An algorithm may be non-scalable, even if it is a CRDT, if its storage or computation requirements explode with size. Complexity theory is an appropriate tool as it studies the inherent costs of some algorithm.

A central concept of distributed systems is consensus [40], which is necessary for many tasks, but constitutes an obstacle to scalability. As CRDTs do not make use of consensus, this allows them to be scalable, but means that the set of tasks solvable using CRDTs is limited.

An well-accepted formalism for distributed computing is the $\pi$-Calculus [23]. It models a system as a set of processes interacting via a process algebra. This makes it possible to prove whether two processes are equivalent, for a given notion of equivalence, using a technique known as bisimulation.

A system is self-stabilising [15] if it eventually recovers from any transient failure. For example, a self-stabilising routing algorithm establishes and maintains routing tables in such a way that if an entry is modified or deleted, it is eventually corrected.

**Chosen methods and anticipated solutions** An important metric for scalability is given by complexity theory. We will study the computational and storage complexity of known CRDTs, and explore whether there are theoretical limits (e.g., upper or lower bounds).

Our working hypothesis is that a task that is solvable using *eventual consensus* is solvable with a CRDT. Eventual consensus is defined as a consensus where processes decide infinitely often, and there is a (monotonically-growing) prefix of the execution over which all processes decide equivalent states. The converse may also hold, i.e., CRDTs and eventual consensus may have exactly the same power. Finding a definite answer to this question would help understanding CRDTs and relate them to existing theory.

We plan to study CRDTs in the $\pi$-Calculus. Defining a canonical CRDT in terms of a process, any process bisimilar to the canonical CRDT is itself a CRDT. This will help understand what can be done using a CRDT and, conversely, their limits, since any process that is *not* bisimilar to the canonical CRDT is not a CRDT.

The concept of eventual consistency leads us to postulate that there is a link between CRDTs and self-stabilisation. In particular, we intend to study whether, and under which conditions, a shared memory can be emulated in a distributed system using a CRDT, for use by a self-stabilising algorithm. This would allow reusing known self-stabilising algorithms.

**Work plan** We shall study the concept of scalability from the complexity perspective, and compare the CRDT class of problems to the consensus class.

We will study a formalisation of CRDTs in the $\pi$-Calculus.

Finally we will study whether self-stabilising algorithms can be used in the context of a distributed system using CRDTs.

| **Deliverables** | **Deadlines** |
|---|---|
| D3.1: CRDTs versus consensus approaches | M12 |
| D3.2: Study of CRDTs in the $\pi$-Calculus | M18 |
| D3.3: CRDTs and self-stabilisation | M24 |

**Success criteria** We expect to precisely understand, using well-established frameworks, what a CRDT is and what it is not; what can be done with a CRDT and what cannot be done, thus integrating the CRDT concept into accepted object taxonomies. We aim to publish these results in significant venues and to leverage them in the other technical tasks.

**Risks** CRDTs might not constitute an interesting class from a theoretical perspective. Complexity of CRDTs may be disappointing. The relation between CRDTs and self-stabilisation might not be established.

## Task 4: CRDT Design

| **Score (leader)** | Regal | UNL | M6 (start) |
|---|---|---|---|
| 17.52 p×m | 14 p×m | 6 p×m | M30 (end) |

**Objectives** This task aims to create a collection of useful CRDT designs and of related techniques.

Some common shared data types – such as set or bag – have naturally commutative operations, but most must maintain invariants that prevent commutativity. We expect that, in a number of cases, it is possible to weaken the invariants in a way that allows operations to be commutative, yet remains useful. We investigate this approach and scalable algorithmic techniques for implementing the required semantics.

Several massive computing infrastructures make use of a key-value map, such as DHTs [10, 36], Amazon's Dynamo [13], Google's Bigtable [12], and Facebook's Cassandra [19]. Preliminary investigation leads us to believe that such a map can be modeled as a CRDT, which will allow such systems to scale even better.

Other possible targets are unordered and ordered trees such as file systems and XML trees. File systems are the base data type of distributed control version systems such as Git or Mercurial. XML trees are used in many applications, including collaborative tools such as Google Wave.

**State of the art** As explained in Section 2.1.3, we were successful in designing sequence CRDTs with insert and delete operations. We achieved this by weakening the usual invariant of sequences: rather than maintaining a strict total order and identifying elements with an index relative to document state, the invariant is based on a partial ordering and we use unique, immutable position identifiers taken from a dense and ordered space. Deletion marks the element as a tombstone. Occasional garbage collection solves the meta-data overhead issue.

Similarly, we were able to design a graph CRDT, the Multilog. Its operations are to create vertices and edges between them. The application using it (Telex) ensures that the graph conforms to some specific structural constraints and grows in a well-defined way. This allows efficient garbage collection and avoids indefinite growth.

**Chosen methods and anticipated solutions** We propose to approach the map CRDT in two phases. In the first, values will be considered atomic, their semantics unknown. Our goal will be to guarantee convergence and consistency between replicas. When concurrent updates to the same key occur, we maintain the set of values assigned. It is the application's responsibility to choose a particular value from the set or to use the whole set.

In the second phase, values have semantics, i.e., type-specific operations that operate on values. For example, a value might itself be a Treedoc. We must therefore study the interaction between multiple levels of CRDT.

The file system data type seems to require consensus, since a directory may not contain more than one element with a given name. This can be circumvented in a model where several elements with the same name are allowed, mapped to a standard file system.

In an XML tree, the attributes of an element are unordered, whereas its children are ordered. The former case can handled with last-writer-wins, or alternatively, as a set (similarly to the map CRDT). The ordering of children might be implemented as a sequence CRDT. However, in addition to inserts and deletes, it is desirable to support a move operation. This is not currently supported and requires more research.

**Work plan** This task starts at Month 6 and continues to Month 30. It is strongly connected to the other technical tasks: Requirements come from Task 2. The specification of CRDTs uses the formalisms developed in Task 3. Actual implementation and evaluation take place in Task 6, whose findings feeding back into this task.

| **Deliverables** | **Deadlines** |
|---|---|
| D4.1: Abstract unordered and ordered tree CRDT | M12 |
| D4.2: File System CRDT | M18 |
| D4.3: XML CRDT | M24 |
| D4.4: Map CRDT | M30 |

**Success criteria** Creating a comprehensive library of CRDT designs and associated techniques. To publish these results in significant venues and to see them used in realistic applications.

**Risks** To obtain results that ensure eventual consistency, but that do not fit users' requirements.

## Task 5: Maintaining strong invariants in the CRDT context

| **Regal (leader)** | Score | UNL | M6 (start) |
|---|---|---|---|
| 16 p×m | 7 p×m | 6 p×m | M30 (end) |

**Objectives** As discussed under Task 3, the properties that a CRDT can ensure are rather weak, since, for instance, a CRDT cannot invoke consensus. Realistically, we cannot expect this to be sufficient (indeed, even Treedoc requires consensus for garbage collection). In some cases, CRDTs might be extended with operations that don't commute, but hopefully are invoked rarely; in others, CRDTs will co-exist with other, non-CRDT data types. Conversely, existing data types will benefit from a redesign whereby most concurrent invocations commute, even if not all. One might name "quasi-CRDT" such coexistence between commutativity and non-commutativity.

A related issue is maintaining higher-level properties. For instance, in co-operative editing, concurrent updates, while technically non-conflicting, may destroy the high-level meaning of the text.[5] Another example: consider a graph CRDT with operations to create and delete vertices and edges; even if every operation maintains a tree structure locally, the merged graph is not necessarily a tree.

This task will examine techniques for maintaining strong and high-level properties. An important use case is XML conformance to a given schema.

**State of the art** XML files are expected to conform to some domain-specific schema or grammar, defined either informally, or with a schema language such as Document Type Definition (DTD), XML Schema Definition (XSD) or a RELAX NG schema. The Harmony system exploits schemas to improve reconciliation [17], but to our knowledge, merging XML files to conform to a given arbitrary schema has never been achieved. It is a very difficult theoretical problem larger than the scope of this project.

Some consensus protocols have been optimised to switch into a more efficient mode as long as operations commute, and switch back to classical operation otherwise [20, 28]. However, as non-commuting operations are allowed to appear at any time, the protocol remains non-scalable.

A different approach, used in Telex [6] and in Treedoc [30] is to move consensus into the background. The consensus protocol is still there (unavoidably, when operations don't commute) but is not a performance bottleneck. However, this may cause applications to roll back (cascading aborts). The CRDT approach does not have this drawback.

In the most recent publication on Treedoc [21], a novel avenue is explored. Instead of switching between modes in time, sub-sets of the system operate in different modes. The *core* is a small set of known, stable machines. All non-core machines are in the *nebula*, which is unlimited, uncontrolled, unknown, and dynamic. Both core and nebula can process commutative operations, but only the core orders the non-commuting ones. At any time, a nebula site can execute a *catch-up* protocol to bring its state up to date with the core. In this way, the non-scalability of consensus

[5] In fact, this applies to all editors: even a sequential system cannot stop co-authors from saying white at the beginning and black at the end. However the problem is exacerbated by fine-grain, lock-free concurrency.

is not an issue. The catch-up protocol adds complexity to nebula sites, but being decentralised it is not a scalability bottleneck, and does not impact normal commutative operation.

In contrast to maintaining the invariant at all times, an alternative is the "eventual conformance" practice, used in software engineering. Consider designing some software artefact that should conform to high-level expectations such as: it should compile, it should pass regression tests, the user interface should be intuitive, etc. However, the artefact goes through many intermediate states that deviate from expectations, either for expediency, or because of a merge; but eventually, some engineer corrects it. We can re-interpret this as follows. The artefact is a CRDT, i.e., updates are never rejected by the system. Occasionally, replicas locally check some high-level invariant (conformance to expectations). In case of a violation, a replica will submit new operations that correct the issue. In the software engineering example, some developer manually generates the correcting operations. In other cases, where the invariant is formalised, one could imagine an automated correction generator. This eventual conformance approach is strongly related to the theoretical concept of eventual consensus studied in Task 3.

**Chosen methods and anticipated solutions** The goal of this task is to explore quasi-CRDTs, to study solutions for XML conformance and to re-engineer some existing abstractions to make most operations commute. We will develop two new techniques to maintaining high-level invariants: the core-nebula approach and eventual conformance.

**Work plan** For the coexistence of commutative and non-commutative operations, we will investigate systematically the approaches outlined above. As we have already investigated two of them (switching modes over time, and moving consensus to the background) in the past, we focus on the core-nebula and the eventual conformance approaches. We will also look at domain-specific methods in the case of XML conformance.

We invented the core-nebula concept to solve a specific problem, rebalancing in Treedoc. In this task, we will investigate its use for additional operations, and will evaluate the impact of such decision in the CRDT concept.

Regarding XML conformance, we will identify, in standard schema definitions, a sub-set of invariants that can be preserved by commutative operations. We expect that in many cases, non-commuting operations are infrequent and non-commuting mode will remain confined, whereas the vast majority of operations occur in commuting mode. For the stronger invariants, we will explore the eventual conformance approach, based on domain-specific grammars, for instance XMI or the Google Wave XML schema.

Two challenges of the eventual conformance approach are to design an automated correction procedure, and to ensure that iterative and concurrent corrections eventually reach a fixed-point. For the former, we might use planning techniques that compute a path (a set of operations) from the current (non-conforming) state to a target (conforming) state. To ensure that fixes converge to a fixed-point, some form of monotonicity may be sufficient.

| **Deliverables** | **Deadlines** |
|---|---|
| D5.1: Core-nebula architecture for Treedoc | M18 |
| D5.2: Quasi-CRDTs | M24 |
| D5.3: Eventual conformance for XML schemas | M30 |

**Success criteria** To be able to maintain high-level invariants, such as XML conformance, in a system based on CRDTs, and to successfully design and implement quasi-CRDTs that exhibit a favorable ratio between commutative and non-commutative operations. To publish these results in significant venues and to see them used in realistic cloud or P2P settings.

**Risks** The coexistence between commuting and non-commuting operations increases complexity; we may find that this cancels out the advantages of CRDTs. Similarly, it carries an overhead that may turn out to be significant. For eventual conformance, we might not be able to generate correction procedures, or correction might not converge.

## Task 6: Applications, implementation and evaluation

| **Score (leader)** | Regal | UNL | M12 (start) |
|---|---|---|---|
| 15 p×m | 16 p×m | 5 p×m | M36 (end) |

**Objectives** In this task we want to implement our proposed algorithms and evaluate their efficiency in comparison with other existing algorithms. We plan to perform a theoretical evaluation in terms of time and space complexities as well as a practical evaluation on human produced traces using various peer-to-peer or collaborative systems deployed on clouds.

**State of the art** Some of the existing CRDT algorithms for linear structures such as Treedoc [21, 30] and Logoot [43] have been implemented and their complexity in terms of time and space have been evaluated with respect to other algorithms. These CRDT algorithms for linear structures have already been evaluated on Wikipedia articles histories and SVN histories.

**Chosen methods and anticipated solutions** The proposed CRDT algorithms for different type structures such as linear structured documents, file systems, XML documents and key-value store will be implemented in Java and then evaluated according to other existing synchronisation algorithms for the respective structures. The evaluation will be performed in two steps: a theoretical evaluation in terms of time and space complexities and a practical evaluation using human produced traces in different collaborative systems that work with the respective structures.

**Work plan** For each designed CRDT an implementation will be performed followed by a theoretical and then a practical evaluation.

CRDT algorithms for linear structured documents will be evaluated on histories of open source software projects developed using distributed version control systems (DVCS) such as Git, Mercurial and Darcs. As DVCS work on file systems, we will replay histories only for merging of source code files that conform to linear structures. In DVCS systems the histories of operations performed by users keep information about concurrency contrary to Wikipedia and SVN ones. Therefore, we will obtain an huge increment compared to the state of art of the evaluations already performed for CRDTs for linear structures.

CRDT algorithms for file systems will be evaluated on complete traces of various projects developed using DVCS systems. The base type of a DVCS system is a file system.

The CRDT algorithms for XML documents will be evaluated on histories of Google Wave. Comparison in terms of complexities on time and space of synchronisation algorithms will be done with other families of algorithms such as OT algorithms used by Google Wave.

The CRDT algorithms for key-value map will be evaluated in the Uniwiki system [26]. Uniwiki system is in fact a system composed of multiple wiki front-ends that fetch and store their data over a DHT. Currently, the replication between DHT peers is managed using WOOT algorithm, but it has been designed to be used with any optimistic consistency maintenance algorithm (OT, CRDT, etc). The Uniwiki system is a good example of a peer-to-peer structured network that can be used in our evaluation of CRDT algorithms.

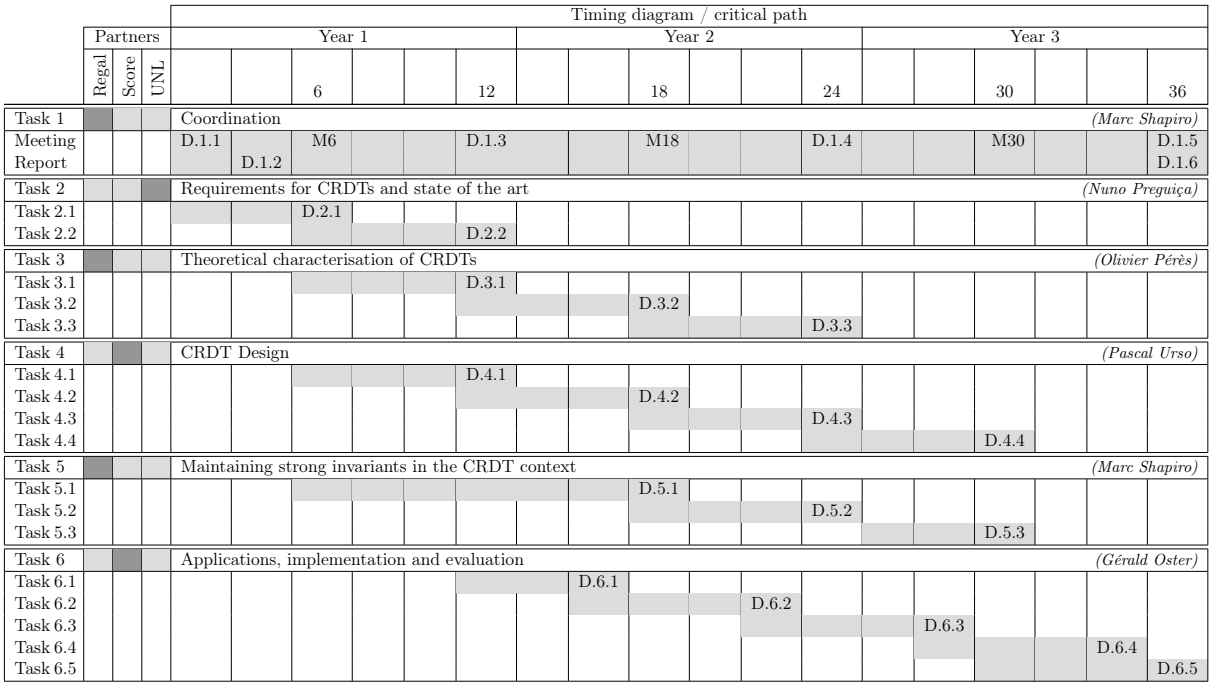
We also plan to perform simulations of the proposed CRDT algorithms on Grid'5000 to test their suitability in large dynamic peer-to-peer systems. Grid'5000 [**?** ] is an infrastructure distributed over nine sites in France. It allows researchers to run experiments on massive-scale parallel and distributed systems. The ConcoRDanT partners have experience with experimenting on Grid'5000.

| **Deliverables** | **Deadlines** |
|---|---|
| D6.1: Implement efficient CRDT for linear structures; evaluate on a DVCS | M16 |
| D6.2: Implement CRDT for file systems; evaluate on a DVCS | M22 |
| D6.3: Implement CRDT for XML; evaluate on Google Wave histories | M28 |
| D6.4: Implement CRDT for key-value map; evaluate on Uniwiki | M34 |
| D6.5: Large-scale simulation of the above on Grid'5000 | M36 |

**Success criteria** The success of this task is the measure of how much CRDTs simplify the task of designing scalable distributed applications, and the performance of these CRDTs. Performance is evaluated both from a complexity viewpoint and experimentally. We aim to publish these results in significant venues.

**Risks** Our current plan is to get traces from Google Wave using a robot. This may turn out to be difficult or forbidden. However, we expect any difficulty to be resolved thanks to our good contacts at Google.

## 3.4 Calendrier des tâches, livrables et jalons / Planning of tasks, deliverables and milestones




**Full list of deliverables and their deadlines**

| | | |
|---|---|---|
| D1.1: | Kick-off meeting | M0 |
| D1.2: | Website | M3 + continuous updates |
| D1.3: | Yearly workshop | M12 |
| D1.4: | Yearly workshop | M24 |
| D1.5: | Final open workshop | M36 |
| D1.6: | Final report | M36 |
| D2.1: | Literature survey | M6 |
| D2.2: | Survey of relevant systems | M12 |
| D3.1: | CRDTs versus consensus approaches | M12 |
| D3.2: | Study of CRDTs in the $\pi$-Calculus | M18 |
| D3.3: | CRDTs and self stabilisation | M24 |
| D4.1: | Abstract unordered and ordered tree CRDT | M12 |
| D4.2: | File System CRDT | M18 |
| D4.3: | XML CRDT | M24 |
| D4.4: | Map CRDT | M30 |
| D5.1: | Core-nebula architecture for Treedoc | M18 |
| D5.2: | Quasi-CRDTs | M24 |
| D5.3: | Eventual conformance of XML schemas | M30 |
| D6.1: | Implement efficient CRDT for linear structures; evaluate on a DVCS | M16 |
| D6.2: | Implement CRDT for file systems; evaluate on a DVCS | M22 |
| D6.3: | Implement CRDT for XML; evaluate on Google Wave histories | M28 |
| D6.4: | Implement CRDT for key-value map; evaluate on UniWiki | M34 |
| D6.5: | Large-scale simulation of the above on Grid'5000 | M36 |

# 4 Stratégie de valorisation des résultats et mode de protection et d'exploitation des résultats / Data management, data sharing, intellectual property and result exploitation

## 4.1 Dissemination of results

As this is a fundamental scientific research project, the main avenue for dissemination of results is publication in the peer-reviewed literature.

The project will organise a final open workshop to disseminate and demonstrate results.

## 4.2 Consortium agreement

To protect the interests of all partners, a Consortium Agreement will be agreed at the beginning of the project. This will settle any remaining organisational issues such as the role of the task leaders, as well as remaining issues of intellectual property and exploitation of results, according to the following principles.

- Pre-existing knowledge and software: The Consortium Agreement will specify reasonable terms under which each partner makes its pre-existing knowledge and software, agreed to be necessary for the pursuit of the project, will be made available to the other partners. Each participant maintains ownership of its preexisting knowledge and software, and remains is sole judge of the measures to be taken for protecting its property.
- Exploitation of results: Results (knowledge and artefacts) resulting from the project are the property of their authors, who may protect their property by patent, copyright, software license, or any other means. Artefacts include source code, data sets, benchmarks, execution traces, and so on. The partners are encouraged to make all artefacts developed in the project widely available (after anonymising any personal data), under a non-restrictive open source license such as BSD or CeCill.
- Publication: Each partner may freely publish its own results, knowledge or artefacts, without permission of other partners. However, partners must not violate the confidentiality or intellectual property of the other partners. Partners will inform all partners in advance about future submissions and publications related to the project.

# 5 Organisation du partenariat / Consortium organisation and description

## 5.1 Description, adéquation et complémentarité des partenaires / Relevance and complementarity of the partners within the consortium

### Partner 1: INRIA Regal

INRIA, the French National Institute for Research in Computer and Control Sciences is an world leader in fundamental and applied research, in the areas of information and communication science and technology. The Institute plays a major role in technology transfer, by research training, scientific and technical information, development, providing expert advice and participating in international programs.

INRIA has eight research centres (Paris-Rocquencourt, Rennes, Sophia Antipolis, Grenoble-Lyon, Nancy, Bordeaux, Lille and Saclay). Its workforce numbers 3,700, of whom 2,900 are scientists, organised in 152 research project-teams. Many INRIA researchers are also professors, whose approximately 1,000 doctoral students work on theses as part of INRIA research project-teams.

Regal is a joint research group of LIP6 and INRIA Paris-Rocquencourt. Regal investigates large-scale distributed systems, and especially P2P architectures. An important focus is large-scale replication in very dynamic settings. We investigate adaptive algorithms, in order to react to changes in the environment and in the application. Here are some of our research areas:

- Data management in large scale configurations: to deploy and locate data, and to manage consistency.
- System monitoring and failure detection: we investigate failure detectors with provable properties in dynamic environments.
- Replication: replication of data improves availability and responsiveness but updates raise the issue of consistency. We have several interests in this area: fault-tolerant replication techniques, optimistic techniques (which allow local updates but cause replication divergence) and adaptive replication. Our research aims to compare and evaluate existing algorithms and to combine their best features into new, high-performance protocols.
- Dynamically-adaptative operating systems

Regal acts as the leader of this project. This is based on Regal's previous experience in distributed systems, especially the design of large-scale, fault-tolerant protocols for the Grid and for Peer-to-Peer systems. Regal has contributed important advances in CRDTs [6, 21, 30].

**Partner 2: LORIA Score**

LORIA, the Lorraine Laboratory of IT Research and its Applications, is a mixed research unit - UMR 7503 - shared by several establishments: CNRS (National Centre of Scientific Research), INPL (National Polytechnic Institute of Lorraine), INRIA (National Research Institute for IT and Robotics), Henri Poincaré University Nancy 1 and Nancy 2 University. LORIA is a Laboratory of more than 450 individuals including more than 150 researchers and teaching-researchers, a third of doctorate students and post-doctorate and a third of engineers, technicians and administrative staffs. LORIA's research teams are involved in more than 40 industrial contracts underway for more than 3 millions euros, and participates in more than 125 co-operation projects with more than 32 different countries

Score team is a joint research group of Henri Poincaré University Nancy 1, Nancy 2 University and INRIA Nancy - Grand Est.

Score team investigates co-operative, distributed, and process-aware Web Information Systems. Its research are organised in two main streams:

- Process Engineering which is interested in process-aware information systems that manage and execute operational processes involving people, applications, and information sources on the basis of process models.
- Collaborative Distributed Systems which is concerned with the development of collaborative systems but with a scientific focus on data consistency in peer to peer architectures. Interactions between these two axes are mainly governed by shared issues, especially on awareness, coordination, and privacy and security management.

Members of Score team have a high expertise in consistency maintenance approaches for optimistic replicated data such as operational transformation approach for instance. These approaches have been applied in the context of several research and industrial projects such as the ANR Xwiki Concerto (peer-to-peer wiki) and the EU FP6 QualiPSo project (collaborative software development platform).

**Partner 3: CITI Distributed Systems Group**

FCT/UNL, Faculdade de Ciências e Tecnologia of Universidade Nova de Lisboa, is a science and technology faculty with close to 500 professors and over 5500 students, located in the Lisbon area. The Center for Informatics and Information Technologies (CITI) is a research institute hosted in the Department of Informatics of FCT/UNL and partially funded by the Portuguese National Science Foundation and by FCT/UNL.

CITI promotes basic and applied research in Computer Science and Informatics. The research directions at CITI cover a wide spectrum of themes, ranging from the foundations and models, programming languages and software architectures, to parallel and distributed computing systems, multimedia, graphics, interaction, and human language technologies and tools. CITI research is in close connection with the graduate and undergraduate teaching mission of the host Department of Informatics.

CITI research team is currently composed by around 40 Senior Researchers, and over 50 graduate students. While the vast majority of the senior researchers are FCT/UNL faculty, CITI has also acted as an attraction pole for researchers from nearby universities.

CITI Distributed Systems Group in involved in two main research directions:

- Transactional Systems, which is concerned with data management support in both multi-core systems and distributed systems. In this context, the team investigates both solutions based on the transactional approach and CRDTs.
- Pervasive Systems, which is concerned with solutions for large-scale distributed and pervasive environments. In this context, the team investigates efficient event dissemination, support for participatory sensing applications and security in sensor and ad-hoc networks.

Members of CITI distributed systems group have a high expertise in data management issues, both in large-scale distributed settings and in cluster and multi-core environments. Over the years, the team has been involved in several nationally-funded projects. It is currently involved in three on-going projects, and several project proposals (national and EU).

## 5.2 Qualification du coordinateur du projet / Qualification of the project coordinator

Marc Shapiro is a Senior Researcher (Directeur de Recherche) at INRIA. He created the SOR research group (Systèmes d'Objets Répartis, Distributed Object Systems), which he led for ten years. During this period he was Principal Investigator for many research projects, both academic and in collaboration with industry. This includes joint research projects with Chorus Systèmes (published at SOSP), Novell, DEC (published at OSDI), Sun Microsystems, Bull, and CNET.

He spent six years and a half at Microsoft Research Cambridge as Senior Researcher, managing the ten-person Cambridge Distributed Systems Group (Camdis). He recruited several young

researchers who moved on to become stars of their field, such as Anne-Marie Kermarrec, Antony Rowstron, Miguel Castro, Youssef Hamidi, and Manuel Costa.

He is now a member of the Regal group and an INRIA employee. Some recently-finished collaborations are: a research grant with Microsoft Research Cambridge, FP6 project Grid4All, ANR project Respire and ARC project Recall.

The Grid4All project was a European FP6 project (2005–2009) with academic partners ICCS, INRIA, KTH, SICS, UPC, UPRC, and industrial partners France Télécom R&D and Antares. Its aim was to democratise access to distributed system technologies and to enable large-scale data sharing for collaborative groups. Marc Shapiro chaired its Scientific Committee and led the data-sharing workpackage.

In 1997–2000, Marc Shapiro was the Principal Investigator of the European Long-Term Research project PerDiS. PerDiS was influenced by real user requirements for multiple, large, complex object databases; a combination of co-operation and isolation; multiple trust and geographical domains. The PerDiS platform was designed for large-scale data sharing based on sophisticated security and memory management algorithms. The PerDiS persistent memory was used for a suite of co-operative CAD applications for the building industry. The PerDiS project included five partners from France, the UK, Portugal and Germany. Two partners came from the construction industry (CSTB and IEZ). It had a total budget of 3 000 000 Ecus and 12 equivalent full-time staff.

### 5.3 Qualification, rôle et implication des participants / Contribution and qualification of each project participant

| | *Partenaire* | *Nom* | *Prénom* | *Emploi actuel* | *Personne × mois* | *Rôle/Responsabilité dans le projet* |
|---|---|---|---|---|---|---|
| 1 | INRIA Regal | Shapiro | Marc | DR1 | 18 | Coordinator, Leader Task 5 |
| | | Makpangou | Mesaac | CR1 | 18 | |
| | | Pérès | Olivier | PostDoc | 18 | Leader Task 3 |
| | | X | | PhD student | 18 | Participant |
| 2 | LORIA Score | Urso | Pascal | MC | 9 | Leader Task 4 |
| | | Oster | Gérald | MC | 9 | Leader Task 6 |
| | | Ignat | Claudia | CR1 | 7.2 | Participant |
| | | Molli | Pascal | MC | 2.52 | Participant |
| | | X | | PhD student | 24 | Participant |
| 3 | UNL LSDCS | Preguiça | Nuno | Ass. Prof. | 9 | Leader Task 2 |
| | | X | | PhD student | 24 | Participant |

## 6 Justification scientifique des moyens demandés / Scientific justification of requested budget

We do not claim any budget for a large equipment funding. Most experiments can be performed on standard desktops or laptops. However, in Task 6, we plan to make use of Grid'5000 for experiments at large scale.

Likewise, the project does not request any subcontracting nor internal expenses.

Our travel requests include the project meetings, which all participants must attend, and partner-to-partner visits that will occur during the project. Two (one-third) of the project meetings will be in Portugal, the others in France.

To ensure a tight and successful collaboration, there will be frequent exchanges and visits with our Portuguese partner. Nuno Preguiça and his PhD student will be invited to France for yearly periods of a few months, and the French participants will visit there for a few weeks.

Finally, our travel budget includes attendance in international conferences, where we will present the results of ConcoRDanT to international research audiences.

**Overall summary of the financial plan**

| | *INRIA Regal* | *LORIA Score* | *Total* |
|---|---|---|---|
| Permanent personnel | 293,688 € | 80,595 € | 347,283 € |
| Non-permanent personnel funded by ANR | 123,120 € | 68,000 € | 191,120 € |
| Non-permanent personnel non-funded by ANR | 0 € | 0 € | 0 € |
| Travel | 16,600 € | 23,960 € | 40,560 € |
| Others expenses | 37,388 € | 40,300 € | 77,688 € |
| Management and structure | 7,084 € | 5,290 € | 12,374 € |
| Environment | 345,950 € | 118,876 € | 464,826 € |
| **Total project cost** | **823,830 €** | **337,021 €** | **1,160,851 €** |
| **Total requested budget:** | **184,192 €** | **137,550 €** | **321,742 €** |

## Partner 1: INRIA Regal

**Équipement / Equipment** We plan to two hefty computers for use in the project.

**Small equipment**

| *Kind* | *Number* |
|---|---|
| Workstation/laptop | 2 |

**Personnel / Staff**

**Permanent personnel**

| *Name* | *Status* | *%* | *Months* |
|---|---|---|---|
| Marc Shapiro | DR1 | 50 | 18 |
| Mesaac Makpangou | CR1 | 50 | 18 |

**Non-permanent personnel funded by ANR**

| *Name* | *Status* | *%* | *Months* |
|---|---|---|---|
| CDD2 | PhD | 100 | 18 |
| CDD3 | PostDoc | 100 | 18 |

*Description of CDD2* - Position: PhD Student

During the first year, the PhD student will work on the state of art and requirements (task 2). In this task, he will closely collaborate with other PhD students from other partners. Next, he

will participate in the definition of new CRDTs (task 4) and their extensions to deal with strong invariants (task 5). Finally, he will contribute to the implementation and the evaluation of the proposed CRDTs (task 6). The PhD will last three years, but Regal only asks for funding for half; the other half will be funded by the team on its own budget. Alternatively, we will share a PhD student co-advised with Score.

*Description of CDD3* - Position: PostDoc

The work of the postdoc hired on this position will mainly focus on task 3. Therefore, a good knowledge in $\pi$-calculus and self-stabilisation systems is required.

**Missions / Travel** The project requires two meetings in Paris, two in Nancy, two in Portugal, and a final workshop at a location to be determined in France. Four participants are planned for each meeting. The travel budget includes funding of the above. We also plan to attend one international conference per year to present research results obtained in this project.

**Travel**

| *Kind* | *Number* |
|---|---|
| National meetings | $\simeq$ 12 |
| National meeting reception | $\simeq$ 8 |
| International meetings | $\simeq$ 8 |
| International conferences | $\simeq$ 3 |

**Autres dépenses de fonctionnement / Other expenses** Regal will invite Prof. Nuno Preguiça of partner UNL for a total period of 6 months distributed over the length of the project.

The final workshop will take place in a venue to be determined, preferably co-located with a national conference. We plan a budget for room rental and associated costs.

**Invitations/visits of/to foreign partner**

| *Kind* | *Number* |
|---|---|
| Room renting for final workshop | 1 |
| Partner 3 invitation (Assistant Professor) | $\simeq$ 6 months |

## Partner 2: LORIA Score

### Équipement / Equipment

Statistically, 30% of the computer inventory is renewed every year. Therefore during this three years project, we expected that approximately one computer per participant will be purchased.

**Small equipment**

| *Kind* | *Number* |
|---|---|
| Workstation/laptop | 4 |

**Personnel / Staff**

**Permanent personnel**

| *Name* | *Status* | *%* | *Months* |
|---|---|---|---|
| Pascal Urso | MCF | 25 | 9 |
| Gérald Oster | MCF | 25 | 9 |
| Claudia Ignat | CR1 | 20 | 7.2 |
| Pascal Molli | MCF | 7 | 2.5 |

**Non-permanent personnel funded by ANR**

| *Name* | *Status* | *%* | *Months* |
|---|---|---|---|
| CDD1 | PhD | 100 | 24 |

*Description of CDD1* - Position: PhD Student

At the first stage, the PhD student will be involved in the task 2 in collaboration with PhD students of other partners. Then, he will participate in the definition of new CRDTs (task 4), their implementation and their evaluation (task 6). He will also contribute to and benefit from the results of tasks 3 and 5. All these tasks are planned to be achieved during a PhD thesis. Score team only requests the ANR funding of two years. The third year will be financed by the team on its own budget.

**Missions / Travel**

We request the approximate funding for about three national meetings attended by four people (a total of 12). This amount includes these planned meetings to which almost all participants have to attend, but also the visits to partner 1 that will occur during the project.

We request the required budget for around two international meetings (attended by four people, a total of 8) which are the two planned meetings (M12 and M30) to the foreign partner (partner 3).

We plan and therefore request budget for approximately six registrations at various international conferences, where we will present the research results obtained in this project.

Finally, to ensure a tight and successful collaboration with the foreign partner (partner 3), we plan invitations and short-period visits to and from this partner. For instance, we will invite his Ph.D student that will be involved in the project for few periods of few months, and participants will make few visits of few weeks. Therefore we request the total budget for 9 months of invitations divided into 5 travels.

**Travel**

| *Kind* | *Number* |
|---|---|
| National meetings | $\simeq 12$ |
| International meetings | $\simeq 8$ |
| International conferences | $\simeq 6$ |

**Invitations/visits of/to foreign partner**

| *Kind* | *Number* |
|---|---|
| Invitation of Partner 3 | $\simeq 3{\times}2$ months |
| Visits to Partner 3 | $\simeq 3{\times}1$ months |

### Partner 3: UNL Large Scale Distributed Computing Systems Group

The Portuguese partner does not claim any budget from ANR.

# 7 Annexes / Appendix

## 7.1 Réferences bibliographiques / References

[] Grid'5000. http://www.grid5000.org/.

[1] Marcos K. Aguilera, Arif Merchant, Mehul Shah, Alistair Veitch, and Christos Karamanolis. Sinfonia: a new paradigm for building scalable distributed systems. In *21st Symp. on Op. Sys. Principles (SOSP)*, volume 41 of *Operating Systems Review*, pages 159–174, Stevenson, Washington, USA, October 2007. Assoc. for Computing Machinery.

[2] Mustaque Ahamad, Phillip W. Hutto, and Ranjit John. Implementing and programming causal distributed shared memory. In *11th Int. Conf. on Distributed Comp. Sys. (ICDCS)*, Arlington, TX, USA, May 1991.

[3] Amazon Web Services Service Health Dashboard. Amazon S3 availability event: July 20, 2008. http://status.aws.amazon.com/s3-20080720.html, July 2008.

[4] B. R. Badrinath and Krithi Ramamritham. Semantics-based concurrency control: beyond commutativity. *Trans. on Database Systems*, 17(1):163–199, March 1992.

[5] Carlos Baquero and Francisco Moura. Using structural characteristics for autonomous operation. *Operating Systems Review*, 33(4):90–96, 1999.

[6] Lamia Benmouffok, Jean-Michel Busca, Joan Manuel Marquès, Marc Shapiro, Pierre Sutra, and Georgios Tsoukalas. Telex: A semantic platform for cooperative application development. In *Conf. Française sur les Systèmes d'Exploitation (CFSE)*, Toulouse, France, September 2009.

[7] Philip A. Bernstein, Vassos Hadzilacos, and Nathan Goodman. *Concurrency Control and Recovery in Database Systems*. Addison-Wesley, 1987.

[8] Ken Birman, Gregory Chockler, and Robbert van Renesse. Toward a Cloud Computing research agenda. *ACM SIGACT News*, 40(2):68–80, June 2009.

[9] Michael Burrows. The chubby lock service for loosely-coupled distributed systems. In *Symp. on Op. Sys. Design and Implementation (OSDI)*, pages 335–350, Seattle, WA, USA, November 2006. Usenix.

[10] Miguel Castro, Manuel Costa, and Antony Rowstron. Performance and dependability of structured peer-to-peer overlays. In *Int. Conf. on Dependable Sys. and Networks*, Firenze, Italy, jun 2004.

[11] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, and Antony Rowstron. Scribe: A large-scale and decentralised application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communication (JSAC)*, 20(8), October 2002.

[12] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems*, 26(2):1–26, 2008.

[13] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. Dynamo: Amazon's highly available key-value store. In *21st Symp. on Op. Sys. Principles (SOSP)*, volume 41 of *Operating Systems Review*, pages 205–220, Stevenson, Washington, USA, October 2007. ACM Sigops, ACM.

[14] B. Devlin, J. Gray, B. Laing, , and G. Spix. Scalability terminology: Farms, clones, partitions, and packs: RACS and RAPS. Technical Report MS-TR-99-85, Microsoft Research, Redmond, WA, USA, December 1999.

[15] Shlomi Dolev. *Self-Stabilization*. MIT Press, 2000.

[16] C. A. Ellis and S. J. Gibbs. Concurrency control in groupware systems. In *Int. Conf. on the Mgt. of Data (SIGMOD)*, pages 399–407, Portland, OR, USA, 1989. ACM SIGMOD, ACM.

[17] J. Nathan Foster, Michael B. Greenwald, Christian Kirkegaard, Benjamin C. Pierce, and Alan Schmitt. Exploiting schemas in data synchronization. *Journal of Computer and System Sciences*, 73(4):669–689, 2007.

[18] Paul R. Johnson and Robert H. Thomas. The maintenance of duplicate databases. Internet Request for Comments RFC 677, Information Sciences Institute, January 1976.

[19] Avinash Lakshman and Prashant Malik. Cassandra: a structured storage system on a P2P network. In *Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 47–47, New York, NY, USA, 2009. ACM.

[20] Leslie Lamport. Generalized consensus and Paxos. Technical Report MSR-TR-2005-33, Microsoft Research, March 2005.

[21] Mihai Leţia, Nuno Preguiça, and Marc Shapiro. CRDTs: Consistency without concurrency control. In *SOSP W. on Large Scale Distributed Systems and Middleware (LADIS)*, Big Sky, MT, USA, October 2009. ACM SIG on Operating Systems (SIGOPS).

[22] Rui Li and Du Li. Commutativity-based concurrency control in groupware. In *Int. Conf. on Collab. Comp.: Networking, Apps. and Worksharing (CollaborateCom)*, page 10, San Jose, CA, USA, December 2005.

[23] Robin Milner. *Communicating and Mobile Systems: The Pi-Calculus*. Cambridge U. Press, 1999.

[24] S. Mishra, L.L. Peterson, and R.D. Schlichting. Implementing fault-tolerant replicated objects using Psync. In *Symp. on Reliable Dist. Sys.*, pages 42–52, Seattle, WA, USA, October 1989. IEEE.

[25] David A. Nichols, Pavel Curtis, Michael Dixon, and John Lamping. High-latency, low-bandwidth windowing in the Jupiter collaboration system. In *Symp. on User Interface and Software Technology (UIST)*, pages 111–120, New York, NY, USA, 1995. ACM.

[26] Gérald Oster, Pascal Molli, Sergiu Dumitriu, and Rubén Mondéjar. UniWiki: A collaborative P2P system for distributed wiki applications. In IEEE Computer Society, editor, *Int. W. on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE)*, pages 87–92, Groningen, The Netherlands, June 2009.

[27] Gérald Oster, Pascal Urso, Pascal Molli, and Abdessamad Imine. Data consistency for P2P collaborative editing. In *Int. Conf. on Computer-Supported Coop. Work (CSCW)*, pages 259–268, Banff, Alberta, Canada, November 2006. ACM Press.

[28] Fernando Pedone and André Schiper. Handling message semantics with generic broadcast protocols. *Distributed Computing Journal*, 15(2):97–107, 2002.

[29] K. Petersen, M. J. Spreitzer, D. B. Terry, M. M. Theimer, and A. J. Demers. Flexible update propagation for weakly consistent replication. In *Symp. on Op. Sys. Principles (SOSP)*, pages 288–301, Saint Malo, October 1997. ACM SIGOPS.

[30] Nuno Preguiça, Joan Manuel Marquès, Marc Shapiro, and Mihai Leţia. A commutative replicated data type for cooperative editing. In *Int. Conf. on Distributed Comp. Sys. (ICDCS)*, pages 395–403, Montréal, Canada, June 2009.

[31] Nuno Preguiça, Marc Shapiro, and Caroline Matheson. Semantics-based reconciliation for collaborative and mobile environments. In *Int. Conf. on Coop. Info. Sys. (CoopIS)*, volume 2888 of *Lecture Notes in Comp. Sc.*, pages 38–55, Catania, Sicily, Italy, November 2003. Springer-Verlag GmbH.

[32] Michel Raynal, André Schiper, and Sam Toueg. The causal ordering abstraction and a simple way to implement it. *Information Processing Letters*, 39(6):343–350, September 1991.

[33] Hyun-Gul Roh, Jin-Soo Kim, and Joonwon Lee. How to design optimistic operations for peer-to-peer replication. In *Int. Conf. on Computer Sc. and Informatics (JCIS/CSI)*, Kaohsiung, Taiwan, October 2006.

[34] Antony Rowstron and Peter Druschel. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In *Symp. on Op. Sys. Principles (SOSP)*, pages 188–201, October 2001.

[35] Yasushi Saito and Marc Shapiro. Optimistic replication. *Computing Surveys*, 37(1):42–81, March 2005.

[36] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Trans. Netw.*, 11(1):17–32, February 2003.

[37] Chengzheng Sun and Clarence Ellis. Operational transformation in real-time group editors: issues, algorithms, and achievements. In *Int. Conf. on Computer-Supported Coop. Work (CSCW)*, page 59, Seattle WA, USA, November 1998.

[38] Chengzheng Sun, Xiaohua Jia, Yanchun Zhang, Yun Yang, and David Chen. Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems. *Trans. on Comp.-Human Interaction*, 5(1):63–108, March 1998.

[39] Douglas B. Terry, Marvin M. Theimer, Karin Petersen, Alan J. Demers, Mike J. Spreitzer, and Carl H. Hauser. Managing update conflicts in Bayou, a weakly connected replicated storage system. In *15th Symp. on Op. Sys. Principles (SOSP)*, pages 172–182, Copper Mountain, CO, USA, December 1995. ACM SIGOPS, ACM Press.

[40] John Turek, T. J. Watson, and Dennis Shasha. The many face of consensus in distributed systems. *IEEE Computer*, pages 8–17, June 1992.

[41] Werner Vogels. Eventually consistent. *ACM Queue*, 6(6):14–19, October 2008.

[42] W. E. Weihl. Commutativity-based concurrency control for abstract data types. *IEEE Trans. on Computers*, 37(12):1488–1505, December 1988.

[43] Stéphane Weiss, Pascal Urso, and Pascal Molli. Logoot: a scalable optimistic replication algorithm for collaborative editing on P2P networks. In *Int. Conf. on Distributed Comp. Sys. (ICDCS)*, Montréal, Canada, June 2009.

## 7.2 Biographies / CV, Resume

### 7.2.1 Marc Shapiro, Directeur de recherche, INRIA Paris-Rocquencourt

**Keywords** Concurrent Programming, Distributed Systems, Replication and Consistency, Distributed Shared Data, Optimistic Replication, Optimistic Concurrency Control, Nomadic Computing, Disconnected Work, Distributed Garbage Collection, Large Scale Distributed Systems.

**Professional experience**

**2005–Today** Senior researcher (*Directeur de recherche*), Regal group (LIP6 & INRIA), Paris (France).

**1999–2005** Senior Researcher, Microsoft Research, Cambridge (UK).

**1986–1999** Senior researcher, INRIA, projet SOR, Rocquencourt (France).

**July–August 1997** Visiting scientist, JINI Project, Sun Research Labs, Chelmsford (MA, USA).

**1993–1994** Visiting Scientist, Cornell University, Ithaca (NY, USA)

**1985** Software engineer, INRIA.

**1984–1985** Software Engineer, GIPSI SM90 (INRIA-Bull-CNET).

**1982–1985** Junior Scientist (*Chargé de Recherches*), Centre Mondial Informatique et Ressource Humaine (CMIRH), Paris (France).

**1982** Assistant Professor, Boston College, Boston (MA, USA).

**1980–1981** Post-doc, Massachusetts Institute of Technology, Cambridge (MA, USA).

**Responsibilities**

**Today** Scientific leader of Grid4All FP6 European project.

**2004–today** Chair and founder of EuroSys, European scientific society for computer systems, European chapter of ACM Sigops.

**1996–2000** Chair and founder of ASF, French chapter of ACM Sigops.

**1996–1999** Principal Investigator of PerDiS European project.

**1995–1999** Vice-chair of ACM Sigops.

**Publications.** **Journals:** Computing Surveys (2005), Computing Systems (1989), Distributed Systems Online (2006), IJAIT (2005), TSI (1984, 1987, 1997). **Book chapters:** LNCS 1752 (2000), IEEE CS Press (1984). **Invited talk:** ICFP (2006). **International conferences:** C++ Conference (1987, 1990), CollaborateCom (2007), CoopIS (2003, 2007), DAIS (2006), DISC–WDAG (1995, 2004), ECOOP (1989, 1998), Euromicro (1980), Euro-Par (2008), FTCS (1981), FTDCS (1995), ICCC (1988), ICDCS (1982, 1986, 1994, 1996), IWMM (1992, 1995), MDM (2004, 2007), OOPSLA (1995, 2004), OSDI (1994), PLDI (1998), PODC (1992, 2001, 2005), POS (1989, 1992, 1994, 1998, 2000, 2000), PPoPP (2006), SIGOPS EW (1986, 1988, 1992, 1995, 2004, 2000), SOSP (1989), SRDS (1991).

**Software and patents** **Patent applications:** Reconcilable and Undoable File System; Probabilistic Scheduling; System-Wide Selective Action Management; Exploiting Dependency Relations in Distributed Decision Making. **Software:** Telex (2007–2008), Joyce (2004-2005), Rufis (2003), IceCube (2000–2003), PerDis (1996–2000), Larchant (1993–1996), SSPC (1990-1994), SOS (1985–1990), COOL-1 (1990–1991), Chorus virtual memory (1986–1988), STL dynamic arrays (1987).

### 7.2.2 Mesaac Makpangou, CR1, INRIA Paris-Rocquencourt

Today, the vast majority of content distributed on the web are produced by web 2.0 applications. Examples of such applications include social networks, virtual universities, multi-players games, e-commerce web sites, and search engines. These applications rely on databases to serve end-users' requests. Hence, the success of these applications/services depends mainly on the scalability and the performance of the database backend.

My current research focus is providing a hosted database replication service. With respect to end-users applications, this service offers an interface to create, to register, and to access databases. Internally, each hosted database is fragmented and its fragments are replicated towards a peer-to-peer network. We anticipate that such a service may improve the performance and the availability of popular web applications, thanks to partial replications of backend databases. Partial database replication on top of a peer-to-peer network raises a number of difficult issues: (i) enforcing replica consistency in presence of update transactions, without jeapordizing the scalability and the performance of the system? (ii) accommodating the dynamic and the heterogenity of a peer-to-peer network with the database requirements?

Currently, we develop a partial database replication protocol, capable to spread out a transaction's accesses over multiple database fragments replicas while guarenteeing that each transaction observes a consistent distributed snapshot of a partially replicated database. We enforce 1-Copy SI for database fragments replicated over a wide area network. Unlike most database replication protocols, ours separates the synchronistation from the certification concerns: A small-scale group of schedulers that do not hold database replicas, cooperate with one antoher to certify update transactions; then, certified transactions are notified to replicas. Futhermore, each replica will be notified only the transactions that impact the that it stores. Thanks to this separation, we avoid waste of computaion resource at replicas that will be used to decide whether to abord or commit an update transaction; Our design choices also permit to reduce bandwidth consumption.

#### Related Publications


- *Mesaac Makpangou*: P2P based Hosting System for Scalable Replicated Databases. EDBT09 International Workshop on Data Management in Peer-to-peer Systems (DAMAP). Saint Petersburg, Russia; march 2009.
- *Ikram Chabbouh and Mesaac Makpangou*: Caching Dynamic Content with Automatic Fragmentation. The Seventh International Conference on Information Integration and Web-Based Applications and Services (IIWAS05); Kuala Lumpur, Malaysia; septembre 2005.
- *Corina Ferdean and Mesaac Makpangou*: A Generic and Flexible Model for Replica Consistency Management. Proceedings of the 1st International Conference on Distributed Computing and Internet Technology (ICDCIT 2004); Bhubaneswar, India; décembre 2004.

- *Ikram Chabbouh and Mesaac Makpangou*: A Configuration Tool for Caching Dynamic Pages. The International Workshop on Web Caching and Content Distribution, Beijing China October 2004.

### 7.2.3 Pascal Urso, Maître de Conférences, University Nancy 1

Pascal Urso, 33 years old, is an assistant professor at the Université Henri Poincaré (UHP) since September 2004, and works in the LORIA laboratory – team SCORE. He received the PhD degree in computer science from the Université de Nice Sophia Antipolis in 2002. Prior to its recruitment at UHP, he worked as a post-doctoral fellow at the University of Namur (FUNDP). His research interests include distributed systems, data replication, collaborative systems, P2P computing and automated theorem proving. Pascal Urso participated to several projects including NoE-Interop, INRIA ARC Recall and EU FP6 Qualipso.

**Selected publications related to the project**

[WUM10] S. Weiss, P. Urso and P. Molli. Logoot-Undo: Distributed Collaborative Editing System on P2P Networks. *IEEE Transactions on Parallel and Distributed Systems*, (to appear).

[WUM09] S. Weiss, P. Urso and P. Molli. Logoot: A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks. In *Proceedings of the 29th IEEE International Conference on Distributed Computing Systems - ICDCS 2009*, pp.404-412, June 2009

[WUM07] S. Weiss, P. Urso and P. Molli. Wooki: A P2P Wiki-Based Collaborative Writing Tool. In *Proceedings of the 8th International Conference on Web Information Systems Engineering - WISE 2007*, pp.503-512, December 2007.

[OMU06] G. Oster, P. Molli, P. Urso and A. Imine. Tombstone Transformation Functions for Ensuring Consistency in Collaborative Editing Systems. In *Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing - CollaborateCom 2006*, November 2006.

[OUM06] G. Oster, P. Urso, P. Molli and A. Imine. Data Consistency for P2P Collaborative Editing. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work - CSCW 2006* pp.259-268, November 2006.

### 7.2.4 Gérald Oster, Maître de Conférences, University Nancy 1

Gérald Oster is an Assistant Professor at University Nancy 1, France. He received his Ph.D. degree in 2005 from the Department of Computer Science at Nancy-Université, France and he was a postdoctoral researcher for one year at ETH Zurich, Switzerland. He participated to several research projects such as RNTL LibreSource, INRIA ARC Recall, ANR XWiki-Concerto, EU FP6 QualiPSo and Wiki 3.0. His domain of research is distributed collaborative editing systems with a focus on optimistic replication in peer-to-peer systems.

**Selected publications related to the project**

[MD09] G. Oster, P. Molli, S. Dumitriu and R. Mondéjar, UniWiki: A Collaborative P2P System for Distributed Wiki Applications. In *Proceedings of the 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, WETICE 2009*, pp.87-92, June 2009.

[IOM07] C.-L. Ignat, G. Oster, P. Molli, M. Cart, J. Ferrié, A.-M. Kermarrec, P. Sutra, M. Shapiro, L. Benmouffok, J.-M. Busca and R. Guerraoui, A Comparison of Optimistic Approaches to Collaborative Editing of Wiki Pages. In *Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing - CollaborateCom 2007*, pp.474-483, November 2007.

[OMU06] G. Oster, P. Molli, P. Urso and A. Imine, Tombstone Transformation Functions for Ensuring Consistency in Collaborative Editing Systems. In *Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing - CollaborateCom 2006*, pp.38-48, November 2006.

[OUM06] G. Oster, P. Urso, P. Molli and A. Imine, Data Consistency for P2P Collaborative Editing. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work - CSCW 2006*, pp.259-268, November 2006.

[IROM06] A. Imine, M. Rusinowitch, G. Oster and P. Molli, Formal design and verification of operational transformation algorithms for copies convergence. *Theoretical Computer Science*, 351(2), pp.167-183, 2006.

### 7.2.5 Claudia Ignat, Chargée de Recherche CR1, INRIA Nancy - Grand Est

Claudia-Lavinia IGNAT obtained a B.Sc. in Computer Science from the Technical University of Cluj-Napoca, Romania and a PhD in Computer Science from ETH Zurich, Switzerland. She is currently a researcher at INRIA-Nancy Grand Est in France. Her research area is collaborative editing with a focus on consistency maintenance over different types of documents such as textual, graphical and XML documents as well as awareness approaches in collaborative environments. She is also currently leading research activities on trust and privacy issues in distributed collaborative editing systems. She participated to several research projects such as INRIA ARC Recall, ANR XWiki-Concerto and Wiki 3.0.

**Selected publications related to the project**

[IN08] C.-L. Ignat and M. Norrie, Multi-level editing of hierarchical documents. *Journal of Computer Supported Cooperative Work - JCSCW*, 17(5-6), pp.423-468, December 2008.

[IPO08] C.-L. Ignat and S. Papdopoulou, G. Oster and M. Norrie, Providing Awareness in Multi-synchronous Collaboration Without Compromising Privacy. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work - CSCW 2008*, pp.659-668, November 2008.

[IO08] C.-L. Ignat and G. Oster, Peer-to-peer Collaboration over XML Documents. In *Proceedings of the 5th International Conference on Cooperative Design, Visualization and Engineering - CDVE 2008*, pp.66-73, September 2008.

[IOM07] C.-L. Ignat, G. Oster, P. Molli, M. Cart, J. Ferrié, A.-M. Kermarrec, P. Sutra, M. Shapiro, L. Benmouffok, J.-M. Busca and R. Guerraoui, A Comparison of Optimistic Approaches to Collaborative Editing of Wiki Pages. In *Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing - CollaborateCom 2007*, pp.474-483, November 2007.

[IN06] C.-L. Ignat and N. Morrie, Draw-Together : Graphical Editor for Collaborative Drawing. In *Proceedings of the ACM International Conference on Computer Supported Cooperative Work - CSCW 2006*, pp.269-278, November 2006.

### 7.2.6 Pascal Molli, Maître de Conférences HDR, University Nancy 1

Pascal Molli graduated from Nancy University (France) and received his Ph.D. in Computer Science from Nancy University in 1996. Since 1997, he is Associate Professor at University of Nancy. He participated in the creation of the INRIA ECOO (Environments for Cooperation) project in 1998 and he was vice-head of the INRIA ECOO Team. From October 2009 to current, He is head of the INRIA SCORE team. Pascal Molli has mainly worked on collaborative distributed systems and focused on problems of consistency of shared data in collaborative environments and awareness models for collaborative editing. He participated to several research projects such as Libresource, Xwiki-concerto and Qualipso.

**Selected publications related to the project**

[WUM10] S. Weiss, P. Urso and P. Molli. Logoot-Undo: Distributed Collaborative Editing System on P2P Networks. *IEEE Transactions on Parallel and Distributed Systems*, (to appear).

[WUM09] S. Weiss, P. Urso and P. Molli. Logoot: A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks. In *Proceedings of the 29th IEEE International Conference on Distributed Computing Systems - ICDCS 2009*, pp.404-412, June 2009

[MD09] G. Oster, P. Molli, S. Dumitriu and R. Mondéjar, UniWiki: A Collaborative P2P System for Distributed Wiki Applications. In *Proceedings of the 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, WETICE 2009*, pp.87-92, June 2009.

[IOM07] C.-L. Ignat, G. Oster, P. Molli, M. Cart, J. Ferrié, A.-M. Kermarrec, P. Sutra, M. Shapiro, L. Benmouffok, J.-M. Busca and R. Guerraoui, A Comparison of Optimistic Approaches to Collaborative Editing of Wiki Pages. In *Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing - CollaborateCom 2007*, pp.474-483, November 2007.

[OMU06] G. Oster, P. Molli, P. Urso and A. Imine, Tombstone Transformation Functions for Ensuring Consistency in Collaborative Editing Systems. In *Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing - CollaborateCom 2006*, pp.38-48, November 2006.

[OUM06] G. Oster, P. Urso, P. Molli and A. Imine, Data Consistency for P2P Collaborative Editing. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work - CSCW 2006*, pp.259-268, November 2006.

[IROM06] A. Imine, M. Rusinowitch, G. Oster and P. Molli, Formal design and verification of operational transformation algorithms for copies convergence. *Theoretical Computer Science*, 351(2), pp.167-183, 2006.

### 7.2.7 Nuno Preguiça, Assistant Professor, FCT/UNL, Portugal

Nuno Preguiça has obtained a PhD in Computer Science from Faculdade de Cências e Tecnologia of Universidade Nova de Lisboa (FCT/UNL). During his PhD he was an intern at Microsoft Research, Cambridge. He is now an assistant professor at FCT/UNL. He also belongs to the direction of the CITI research centre hosted at FCT/UNL. His research interests lie in the broad area of replicated data management. He is currently leading two national-funded projects on support for Byzantine faults in database systems (Byzantium - finishing soon) and on using replication in multi-core systems for improving performance and reliability (RepComp).

**Selected publications related to the project**

[LPS09] M. Letia, N. Preguiça and M. Shapiro, CRDTs: Consistency without concurrency control. In *Proceedings of the 3rd ACM SIGOPS International Workshop on Large Scale Distributed Systems and Middleware - LADIS 2009*, October 2009.

[PMS09] N. Preguiça, J. Marquès, M. Shapiro and M. Letia, A commutative replicated data type for cooperative editing. In *Proceedings of the* $29^{th}$ *IEEE International Conference on Distributed Computing Systems - ICDCS 2009*, pp. 395-403, June 2009.

[PRH08] N. Preguiça, R. Rodrigues, C. Honorato and J. Lourenço, Byzantium: Byzantine-Fault-Tolerant Database Replication Providing Snapshot Isolation. In *Proceedings of the Fourth Workshop on Hot Topics in System Dependability*, December 2008.

[ABP07] P.-S. Almeida, C. Baquero, N. Preguiça, and D. Hutchison, Scalable Bloom Filters. *Information Processing Letters (Elsevier)*, 101(6), pp. 255-261, 2007.

[PMD06] N. Preguiça, J. Legatheaux Martins, H.-J. Domingos and S. Duarte, Supporting multi-synchronous groupware: data management problems and a solution. *International Journal of Cooperative Information Systems - IJCIS*, 15(2), pp. 229-258, 2006.

[PSM03] N. Preguiça, M. Shapiro, and C. Matheson, Semantic-based reconciliation for collaboration in mobile environments. In *Proceedings of The Eleventh International Conference on Cooperative Information Systems - CoopIS 2003, (Springer, LNCS 2888)*, pp. 38-55, November 2003.

## 7.3 Implication des personnes dans d'autres contrats / Involvement of project participants to other grants, contracts, etc...

| *Part.* | *Nom de la personne participant au projet* | *Pers. × mois* | *Intitulé de l'appel à projets, Source de financement (Montant attribué)* | *Titre du projet* | *Nom du coordinateur (affiliation)* | *Dates début–fin* |
|---|---|---|---|---|---|---|
| 1 | M. Shapiro | 3 | Google Research Award (33 K€) | Consistency w/o Concurrency Control | M. Shapiro (*INRIA Regal)* | 2009–2010 |
| | M. Shapiro | 9 | ANR Verso 2009 (152 K€) | Prose | A. Chaintreau (*Thomson)* | 2009–2012 |
| 2 | C. Ignat | 5 | Ministère Économie, Industrie, et Emploi (152 K€) | Wiki3 | F. Mancinelli (*XWiki SAS)* | 2010–2011 |
| | C. Ignat | 5 | Europe FP6 | QualiPSo | M. Melideo (*Engineering)* | 2006–2010 |
| | G. Oster | 3 | Ministère Économie, Industrie, et Emploi (152 K€) | Wiki3 | F. Mancinelli (*XWiki SAS)* | 2010–2011 |
| | P. Molli | 3 | Ministère Économie, Industrie, et Emploi (152 K€) | Wiki3 | F. Mancinelli (*XWiki SAS)* | 2010–2011 |
| | P. Molli | 3 | Région IdF (101 K€) | Coclico | C. Remy (*Bull)* | 2009–2011 |
| 3 | N. Preguiça | 9 | FCT/MCTES (130 K€) | Byzantium | N. Preguiça | 2008–2010 |
| | N. Preguiça | 9 | FCT/MCTES (90 K€) | RepComp | N. Preguiça | 2010–2012 |