

# Dynamic Data Replication and Consistency in Mobile Environments

Abdelkrim Beloued, Jean-Marie Gilliot,  
Maria-Teresa Segarra  
GET-ENST Bretagne Département informatique  
Technopole Brest Iroise CS 83818  
29238 Brest Cedex 3, France  
{abdelkrim.beloued, jm.gilliot,  
mt.segarra}@enst-bretagne.fr

Françoise André  
IRISA/université de Rennes 1  
Rennes, France  
francoise.andre@irisa.fr

## ABSTRACT

In this article, we investigate the usage of context-aware replication in mobile environments. We present the problems that can occur in such situations and show that existing replication techniques are not well adapted to this problem. For achieving this replication aspect, we propose a context-aware replication system that adapts dynamically the data replication to changes in context informations. We present an internal and external view of this system and the expected adaptation.

## Categories and Subject Descriptors

C.2.4 [COMPUTER-COMMUNICATION NETWORKS]: Distributed Systems; D.2.11 [SOFTWARE ENGINEERING]: Software Architectures; D.4.7 [OPERATING SYSTEMS]: Organization and Design

## General Terms

Design, Management

## Keywords

Replication, consistency, adaptation and reconfiguration, context description

## 1. BACKGROUND AND MOTIVATION

Data replication is a technique that was initially used in traditional distributed environments to increase data availability and improve system performance. These environments are characterized by a fixed infrastructure where the user uses fixed machines that have sufficient resources and are permanently connected to the network. These characteristics are not verified in mobile environments. In this case,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*2nd International Doctoral Symposium on Middleware '05*, November 28-December 2, 2005 Grenoble, France  
Copyright 2005 ACM 1-59593-267-4/05/11 ...\$5.00.

user devices such as PDAs and mobile phones have limited capacity in terms of memory space, disk space and processor capacity. These limitations may prevent the replication system from creating and placing the replicas on the user device.

In the mobile environment, the user may also change his device to access a service. The diversity of devices and consequently the context of use of a service or an application must be taken into account. Indeed, there is a change in device capacities such as screen size, storage capacity and battery power, and its network parameters such as the type and bandwidth. Thus the mode of application must be adapted to each context.

In short, mobile environments are characterized by a frequent change in their resources which comes from various sources such as the nature of the wireless network itself, the mobility of users and multi-terminal accesses.

This change may influence data replication because the creation of and access to these data may need a set of resources. For example, confidential data like a credit card number may not be replicated and exchanged across non-secure nodes and links. Thus, variation in the level of security may prevent the user from accessing this data. So, a traditional system is not able to satisfy the client's request. Consequently, to ensure service continuity, the replication system functionalities like creation, placement, read, write and consistency operations must be adapted to all variations in resources that data may need.

In our work, we are interested in this aspect of replication and we propose a context-aware replication system. This article presents the internal and external architecture of this system and is organised as follows: Section 2 presents different replication techniques and shows their limitations in mobile environments. Section 3 presents our replication system. Section 4 concludes this article and presents our future work.

## 2. RELATED WORK

Several fields of computer science are interested in data replication techniques. Among these are fault-tolerant services, distributed databases, distributed files, content delivery networks, distributed objects, and distributed shared memories. Different solutions for data replication and consistency management are proposed according to the con-

straints and objectives of each field.

For fault-tolerant services, the appropriate system for achieving fault tolerance is the group communication system that is introduced in [2](see also [3]). To ensure fault tolerance, the correctness criteria of linearisability and sequential consistency must be satisfied. Two principal techniques are proposed in order to satisfy these constraints: passive replication and active replication [3] [8] [15] [18]. In passive replication, clients communicate with a primary replica that propagates update messages to other replicas. In active replication, clients communicate by multicast with all replicas. Other hybrid techniques like semi-active [9] and semi-passive [4] replication combine the previous techniques.

In the database field, the correctness criterion is one-copy serialisability [1] which means that an execution of a set of transactions on a replicated database is equivalent to a serial execution of these transactions on a non-replicated database. Several replication strategies are proposed in the literature to ensure this criterion. [7] [18] have categorised these strategies in four classes according to: the update propagation approach and the update ownership approach. The propagation strategy can either be an eager propagation or a lazy propagation. In case of eager propagation, updates are applied to all replicas as part of one atomic transaction. In the lazy replication, the update messages are asynchronously propagated to other replicas after the transaction commit. The ownership strategy can be a master or a group ownership. In the former case, updates are first applied to the primary copy and then to other copies. In the latter case, any copy can update other replicas. Consequently, we can distinguish four classes of replication strategies: eager group replication, eager master replication, lazy group replication, and lazy master replication.

Some existing systems try to ensure a high data availability like Bayou and Coda, but these systems establish a weak data consistency. Bayou [5] is a mobile database replication platform on which developers can build collaborative applications. Bayou adopts a flexible replication strategy (read any/write any) that allows a user to access the data from any node. An anti-entropy protocol is used for update propagation between weakly consistent replicas. This protocol is based on pair-wise communication. Bayou allows the detection and the resolution of update conflicts in an application-specific manner. In order to do this, dependency checks and merge procedures are used. Coda [12] [3] is a file system that replicates each file on a group of servers and manages the network disconnection. This allows a mobile user to work in disconnected mode by accessing a local cache. The replication strategy that is used in Coda is a variant of a read-one/write-all approach. Coda also allows the detection and the resolution of update conflicts by means of Coda version vector (CVV) and Coda repair tool.

Some previous works on replication consider some environment parameters that are related to the system performance which may indeed be influenced by the change in mobile environment resources. For example, the bandwidth degradation and the change in user location can considerably increase the response time and the network overhead. Among these studies are [11] [10] that take into account, at replica placement time, network topology information and user location in order to reduce the client access latency and the network overhead. [19] propose a dynamic replication scheme which is based on users' mobility schedules,

access behavior and read/write patterns. [16] propose a generalized framework for replica hosting system that automatically adapts the replication scheme to variations in some parameters which are related to the system performance and are classified as controllable and uncontrollable parameters. Controllable parameters are those whose value can be controlled by the system e.g, the number of replicas and their location. Uncontrollable parameters are those which cannot be directly controlled by the system like client request rates and update rates for Web documents. [17] propose replica creation in mobile devices to allow the operation in disconnected mode and/or to avoid the use of weak bandwidth links.

However, these works do not take into account the nature of data. Indeed, according to its nature, data may need a set of resources such as network bandwidth and storage space in order to be replicated. Other environment information like node security, device type and user activity may also be essential for replicating or performing any operation on this data. For example, for the banking application, data is confidential. That is, to create and place this data, we need a high security level for nodes and links. The change in some of this information might not influence the performance of the system and the previous works cannot therefore provide a satisfying solution to replication.

This limited context information management has motivated us to study this replication aspect. Thus, we propose a context-aware replication system, the architecture of which is presented in the following section.

### 3. EXPECTED CONTRIBUTION

#### 3.1 Execution context

There are many definitions of context. Among them, Schilit et al. [13] consider three important aspects of context: where you are, who you are with, and what resources are nearby. Context can thus include lighting, network connectivity, communication bandwidth, etc. Dey et al. [6] define the context as *any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves.*

Context information is first sensed by sensors. Next, it is handled by a context-aware infrastructure [14] [6] that provides a high-level context information to an application. All of these steps are outside the scope of this paper.

In our work, we need two types of context: required context and provided context.

The required context consists of data constraints and user preferences. According to its nature, data may need a set of resources like storage space and bandwidth. So, the application designer defines a set of constraints that have to be satisfied in order to create data replicas or perform any operation on these replicas such as read/write operations and consistency.

The provided context consists of all software properties such as the operating system and programming language, hardware properties like available processors and storage space, and physical environment properties like meeting room state. This type of context may also include the user profile, like his location and daily activities.

Our replication system takes into account this information

and adapts a replication scheme to it.

### 3.2 Internal view of our replication system

The principal functionalities of a replication system are: replica creation, replica placement, read/write operation and replica consistency. Consequently, our system contains three principal modules: a replica planner, a localisation manager and a consistency manager as shown in Figure 1.

The replica planner is responsible for the creation and placement of replicas on nodes. Next, the localisation manager locates replicas for read/write operations and then performs these operations. Finally, the consistency manager ensures replica consistency by exchanging update messages after each write operation and resolving update conflicts.

The other modules and the system functionality will be presented progressively in the following sections.

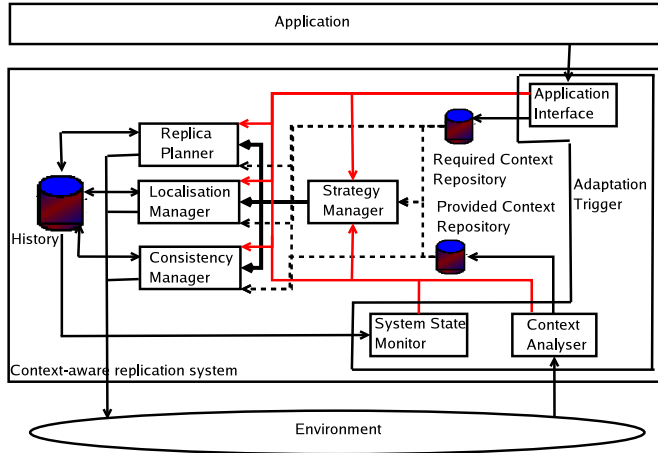


Figure 1: Internal view of our replication system

Our replication system takes into account the execution context and adapts its functionalities to changes in this context. We consider three dimensions of adaptation according to the adaptation trigger (adaptation to context variations, adaptation to system state variations), the adaptation moment (adaptation at the application deployment time i.e, configuration, adaptation at the application run time i.e, reconfiguration), and the adaptation object (replication scheme adaptation, replication strategy adaptation).

Thus, we have to deal with the replication scheme and replication strategy configuration and reconfiguration as shown in Table 1.

	Configuration	Reconfiguration
Replication scheme	Replication scheme configuration	Replication scheme reconfiguration
Replication strategy	Replication strategy configuration	Replication strategy reconfiguration

Table 1: Adaptation classes

#### 3.2.1 Replication scheme configuration

The replication scheme is represented by three plans: a placement plan, a localisation plan and a consistency plan.

These plans are provided by the replica planner, the localisation manager and the consistency manager respectively.

The placement plan indicates the placement of each replica on nodes. Next, the localisation plan indicates for each user a set of nodes and links where he can reach each replica. Finally, the consistency plan indicates for each replica a set of nodes and links where an update can be propagated from this replica to other replicas.

The replication scheme is provided at the application deployment time according to the current context and the current system state. This replication scheme is then stored in the history. The placement plan is immediately projected onto the environment. The localisation and consistency plans are used at the read/write time.

#### 3.2.2 Replication scheme reconfiguration

In order to monitor the execution context and the system state, we propose adaptation trigger modules (Application Interface, Context Analyser, and System State Monitor). The application interface and the context analyser detect the pertinent change in required context information and provided context information respectively. Next, they store this change in the required context repository and the provided context repository. Finally, they notify different modules (replica planner, localisation manager and consistency manager) of this change. These latter provide a replication scheme that is most adapted to the change in context by modifying their corresponding plans that are stored in the history. For example, if the bandwidth is decreased, the system changes some replica locations in order to avoid the use of weak bandwidth links.

The basic objective of the replication technique is to improve the performance and to increase the data availability and consistency. To reach these objectives, our replication system monitors its own state. This state is represented by system performance parameters, data availability parameters, and data consistency parameters. For example, the response time is a parameter characterizing the system performance. Each parameter is measured by a metric. Different parameters are evaluated by the system and stored in the history. The system state monitor detects the pertinent change in these parameters and notifies the replica planner, localisation manager and consistency manager of this change. These modules modify the replication scheme in order to redress the system state. For example, if the response time increases, the replica planner modifies some replica locations in order to reduce this parameter.

#### 3.2.3 Replication strategy configuration

Our replication system adapts the replication strategy to context or system state variations. For example, the system changes its strategy from a pessimistic strategy to an optimistic one in order to improve the data availability. This adaptation is based on a set of rules that specify the strategy application conditions. To manage this adaptation, we propose a strategy manager that ensures system consistency and implements a new strategy.

The strategy configuration is carried out at the application deployment time. Based on the current context and the current system state, the strategy manager chooses the most adapted replication strategy and implements it in the replica planner, localisation manager, and consistency manager.

#### 3.2.4 Replication strategy reconfiguration

The replication strategy reconfiguration is carried out at application run time. After receiving the notification from the adaptation trigger modules, the strategy manager chooses the adapted strategy, ensures the system consistency and implements this strategy in some or all modules (replica planner, localisation manager, and consistency manager).

### 3.2.5 Adaptation Process

To summarize, at the application deployment time, the current context and system state are analysed by adaptation trigger modules. Based on this analysis, the strategy manager chooses the most adapted replication strategy and implements it in the replica planner, localisation manager, and consistency manager. Then these modules provide a replication scheme that is adapted to this context as shown in Figure 2.

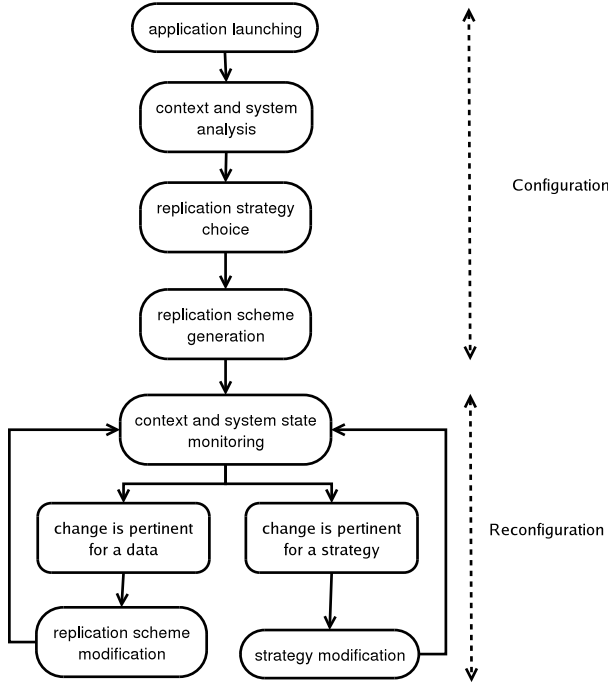


Figure 2: Adaptation process

At the application run time, adaptation trigger modules monitor context information and system state parameters and detect the pertinent change in this information. If this change is pertinent for a replication strategy then these modules notify the strategy manager. Otherwise, if it is pertinent for data then trigger modules notify the replicas planner, localisation manager and consistency manager. In the former case, the strategy manager chooses the replication strategy that is adapted to new information and ensures the system consistency. In the latter case, the replica planner, localisation manager and consistency manager modify their plans that are stored in the history.

### 3.3 External view of our replication system

Our replication system interacts with the application in order to communicate data to be replicated and their resource constraints as shown in Figure 3. It also interacts with the execution environment in order to collect its re-

sources and to project the replication scheme onto this environment.

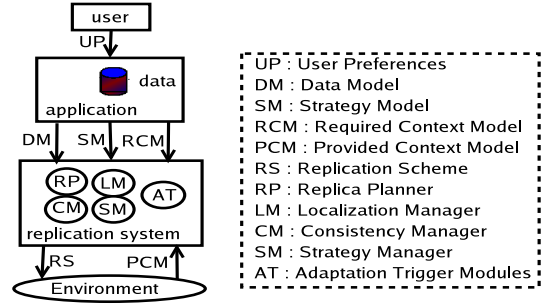


Figure 3: External view of the replication system

These interactions are carried out according to a set of formats (meta-models) that are defined by the system designer.

First, the application designer must define and describe the data to be replicated. This description allows the application to treat the data and to communicate them to the replication system. This latter can be based on a standard data format like object and component data, or a specific format that describes the data structure.

The application designer must also describe the data constraints before communicating them to the replication system. This description is carried out according to a well defined format that specifies the manner in which different replication system modules can understand data constraint semantics. This format is defined by the system designer and represents a meta-model for the required context description.

Provided context information must also be described in order to exchange and analyse them. This description is based on a meta-model for the provided context description.

Replication strategies and their corresponding rules have to be described in order to adapt them to execution context and system state variations.

Finally, the replication scheme also has to be described in order to apply the replication strategy that is based on this scheme.

## 4. FUTURE WORK AND CONCLUDING REMARKS

We have presented in this article the architecture of our replication system. This architecture comprises of four principal components for adapting data replication and consistency to the context: for adapting the replication strategy, the replica placement, read/write operations, and the data consistency. Each component is based on context information and system state parameters in order to choose the replication strategy or generate the corresponding plan. In this paper, we have identified the necessary meta-models to carry out the different interactions of our system: the meta-model for the description of data, strategy and context. We have also identified necessary adaptation classes. Our replication system allows the adaptation and the reconfiguration of the replication scheme and strategy.

Our future work involves in defining each element mentioned above. That is, we have to define each meta-model, system module and adaptation approach.

## 5. REFERENCES

- [1] P. Bernstein, V. Hadzilacos, and N. Goodman. *Concurrency Control and Recovery in Database Systems*. Addison-Wesley, 1987.
- [2] K. P. Birman. The process group approach to reliable distributed computing. *Commun. ACM*, 36(12):37–53, 1993.
- [3] G. Coulouris, J. Dollimore, and T. Kindberg. *Distributed systems : concepts and design*. Pearson Education, third edition, 2001.
- [4] X. Défago, A. Schiper, and N. Sergent. Semi-passive replication. In *SRDS '98: Proceedings of the The 17th IEEE Symposium on Reliable Distributed Systems*, pages 43–50, Washington, DC, USA, 1998. IEEE Computer Society.
- [5] A. J. Demers, K. Petersen, M. J. Spreitzer, D. B. Terry, M. M. Theimer, and B. B. Welch. The bayou architecture: Support for data sharing among mobile users. In *Proceedings IEEE Workshop on Mobile Computing Systems and Applications*, pages 2–7, 1994.
- [6] A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Workshop on the What, Who, Where, When, Why and How of Context-Awareness (CHI 2000)*, April 2000.
- [7] J. Gray, P. Helland, P. O’Neil, and D. Shasha. The dangers of replication and a solution. In *SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, pages 173–182, New York, NY, USA, 1996. ACM Press.
- [8] R. Guerraoui and A. Schiper. Software-based replication for fault tolerance. *IEEE Computer*, 30(4):68–74, April 1997.
- [9] D. Powell, M. Chéréque, and D. Drackley. Fault-tolerance in delta-4\*. *SIGOPS Operating Systems Review*, 25(2):122–125, 1991.
- [10] M. Rabinovich, I. Rabinovich, R. Rajaraman, and A. Aggarwal. A dynamic object replication and migration protocol for an internet hosting service. In *Proc. of the IEEE International Conference on Distributed Computing Systems*, pages 101–113, 1999.
- [11] P. Radoslavov, R. Govindan, and D. Estrin. Topology-informed internet replica placement. In *Proceedings of the Sixth International Workshop on Web Caching and Content Distribution, Boston, MA, 2001*.
- [12] M. Satyanarayanan, J. J. Kistler, P. Kumar, M. E. Okasaki, E. H. Siegel, and D. C. Steere. Coda: A highly available file system for a distributed workstation environment. *IEEE Trans. Comput.*, 39(4):447–459, 1990.
- [13] B. N. Schilit, N. I. Adams, and R. Want. Context-aware computing applications. In *the Workshop on Mobile Computing Systems and Applications*, pages 85–90. IEEE Computer Society, December 1994.
- [14] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. V. Laerhoven, and W. V. de Velde. Advanced interaction in context. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 89–101, London, UK, 1999. Springer-Verlag.
- [15] F. B. Schneider. Implementing fault-tolerant services using the state machine approach: a tutorial. *ACM Comput. Surv.*, 22(4):299–319, 1990.
- [16] S. Sivasubramanian, M. Szymaniak, G. Pierre, and M. V. Steen. Replication for web hosting systems. *ACM Comput. Surv.*, 36(3):291–334, 2004.
- [17] M. T. Segarra and F. André. Mfs: a mobile file system using generic system services. In *SAC '99: Proceedings of the 1999 ACM symposium on Applied computing*, pages 419–420, New York, NY, USA, 1999. ACM Press.
- [18] M. Wiesmann, F. Pedone, A. Schiper, B. Kemme, and G. Alonso. Understanding replication in databases and distributed systems. In *Proceedings of 20th IEEE International Conference on Distributed Computing Systems (ICDCS'2000)*, pages 264–274. IEEE Computer Society, April 2000.
- [19] S. Wu and Y. Chang. An active replication scheme for mobile data management. In *Proceedings of the Sixth International Conference on Database Systems for Advanced Applications (DASFAA '99)*, pages 143–150. IEEE Computer Society, 1999.