Collaborative Negotiation to Resolve Conflicts among Replicas in Data Grids

Ghalem Belalem and Belabbes Yagoubi
Department of Computer Science, Faculty of Sciences, University of Oran
Algeria

1. Introduction

In Data Grids, data replication is a fundamental mechanism widely used to increase data availability, improve performance, balancing the load among nodes of the grid and support the fault tolerance components. The replication management and its implementation are not simple tasks and produce other problems, like consistency management of replicas. One of the concerns major in the consistency management approaches called optimistic, it is the conflicts resolution among replicas. One of the main problems encountered in the use of replication, the management of consistency among the replicas. Replication techniques are used to provide multiple critical copies and to maintain them. In coherent state, they improve the overall system availability and performance.

The main objective of a replica consistency approach is to avoid or even reduce the inconsistency between replicated data. Many current applications can barely tolerate a certain degree of contradiction between replicas where the strong consistency is not a condition, for examples in the approximate readings from meteorological sensors often suffice when performing predictive modelling of weather conditions, the network security applications or in video conferencing applications.

Our contribution consists of the proposal for a model, for consistency management of replicas in data grids, which combines at the same time the pessimistic approaches, which support the quality of service (QoS), and the optimistic approaches which are focused on the improvement performance. Our effort in this contribution, aims to resolve conflicts between the replicas by using a collaborative negotiation between representatives of the nodes in the data grid. We try to show in this work that trading is influenced by the load balancing of the grid.

Thereafter, we organize our paper as follows: Section 2 gives an overview some work for the resolution of conflicts between replicas, section 3 will be reserved for a brief description of the management techniques used consistency of replicas, the Section 4 presents our approach to consistency management approach that combines optimistic and upbeat and then we propose in Section 5, a process for resolving conflicts encountered between replicas is based on economic models of the real market. Section 6 allows us to study the influence of load balancing on quality replicas. The experimental study of our approach is presented in Section 7, we compare the behaviour of our proposal in comparison with conventional techniques, in a first phase, we try to show that our approach reduces significantly the divergences and conflicts, in a second part we will present the effect of balancing the quality of replicas. We conclude this paper with a summary and a set of perspectives.

2. Related work

One of the problems hard in the consistency management is the choice of the technique of reconciliation to make converge the divergent replicas. The reconciliation is the activity to detect, control and solve conflicts, in order to converge towards the same state. Several work proposed different techniques to produce a new coherent value of the counterparts of the same object. Coda (Kistler and Satyanarayanan 1992) was among the first hoarding systems, using user profiles to decide what to hoard and requiring user intervention for conflict resolution. In the system Bayou (Petersen et al. 1996) conceived for the collaborative applications in the mobile environments and using the optimistic replication to weak coherence, the coherence of the replicas east guarantees thanks to an epidemic diffusion of the updates between the waiters. If a conflict emerges the waiters in Bayou proceed by fusion of the requests of the writings according to a quite selected scheduling. In the system IceCube (Kermarrec et al. 2001) uses the reconciliation based on semantic specificities of the application and the intention of the user. He amalgamates the newspapers compared to recorded operations and of which the goal to minimize the conflicts. In the work presented in (Molli et al. 2002; Vidot et al. 2000), authors use the technology of the operational transforms by exploiting the semantic properties of the operations such as causality for serializability in order to lead to the coherence of a divided object. Instead, the work presented here is based on less constraining assumptions about the semantic of data and thus their ability to reconcile inconsistent data.

3. Replication and consistency management

Replication techniques are used to provide multiple critical copies and to maintain them. In coherent state, they improve the overall system availability and performance. In splitting of replication advantages, there are many problems that must be resolve like (Belalem & Bouhraoua, 2007; Belalem & Slimani, 2007; Gray et al., 1996; Xu et al., 2002):

How do we select and estimate the metrics for taking replication decisions?
When do we replicate a given object?
Where do we place the replicas of a given object?
How do we ensure consistency of all replicas of the same object?
How do we route client requests to appropriate replicas?

Among these problems, the main critical concerns the consistency problem that needs to maintain the data consistency between a set of replicated data distributed among a set of computer (Saito & Shapiro, 2005; Goel et al., 2005; Cameron et al. 2004). The main objective of a replica consistency approach is to avoid or even reduce the inconsistency between replicated data. Many current applications can barely tolerate a certain degree of contradiction between replicas where the strong consistency is not a condition, for examples in the approximate readings from meteorological sensors often suffice when performing predictive modelling of weather conditions, the network security applications or in video conferencing applications (Belalem & Slimani, 2007; Dorcey, 1995).

3.1 Pessimistic approach

The Pessimistic approach prohibited any access to a replica unless it is provably up to date. This makes the users believe that they have only one consistent copy. The main advantage of this approach is that all replicas converge at the same time, a fact which permits to guarantee a high consistency. Hence, any problem of divergence could be avoided. This type of approach is well adapted to small and middle scale systems but becomes very complex when adapted to large scale systems. Thus, we can raise three major drawbacks of this approach (Saito & Shapiro, 2005):
it is very badly adapted to uncertain or unsteady environments, such as the mobile systems, or grids with high rate of change;
it cannot bear the updating cost when the degree of replication is very high.

3.2 Optimistic approach

Unlike the pessimistic approach, this approach allows acceding to any replica and at any time. In this way, it is then possible to reach any replica even if it is not necessarily coherent. So, this approach tolerates a certain divergence among replicas. On the other hand, it requires a phase to detect divergence among replicas then a phase to correct this divergence by converging the replicas toward a coherent state. Although it does not guarantee a high consistency as in the pessimistic case, it has, however, a certain number of advantages. They are mainly summarized as follows (Olston & Widom, 2005; Saito & Shapiro, 2005):

- since it admits a divergence among replicas, this approach can be adapted to uncertain or mobile environments;
- it does not block the accesses to replicas;
- it allows the passage to scale by accepting a high number of replicas.

4. Consistency management of replicas in Data Grid

Our main contribution in this present work is to propose a service of consistency management for replicas in a Data Grid environment. The service combines the pessimistic and optimistic approaches in order to increase service quality replicas and improve the performance of the Data Grid. The service proposed is supported by a model hierarchical and distributed to two levels (see Figure 1), which enables him to adapt to the scalability, to be flexible and to allow the tolerance of certain faults. We consider a grid as a collection of distributed collections of Computing Elements (CE's) and Storage Elements (SE's). These elements are linked together through a network to form a Site or a Cluster. Sites are in turn linked together to form a grid. Replicas are stored on Storage Elements and are accessible from Computing Elements.

In our work, the data grid consists of two levels:
- Level 0: In this level, the management of replicas of data within a site is performed using one or more replicas managers (RM) as replication strategies (cite!!!). Two strategies can be

used: the replication strategy type uni-master where the master is the replica manager (RM) and the multi-masters where the site is managed by several managers replicas.

- Level 1: This level consists of a set of representatives (virtual nodes). Each representative of the level 1 is a site of level 0. For each site, a representative is elected from the manager replicas.

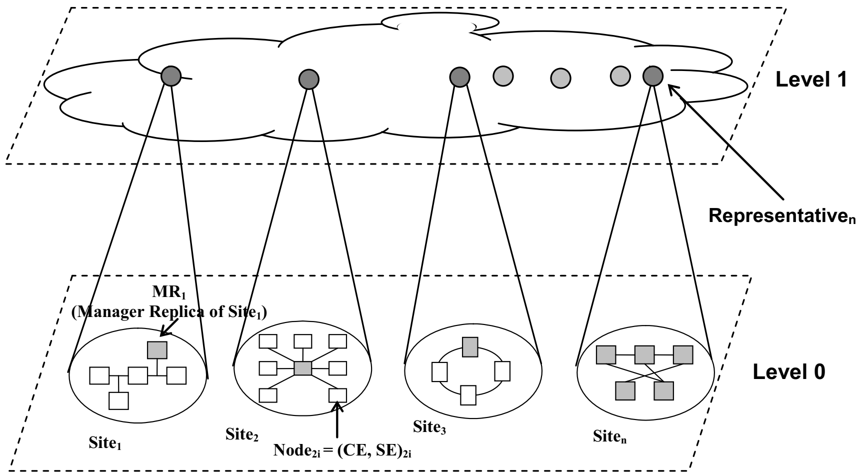

Fig. 1. Proposed Model for Consistency Management

The proposed service for consistency management uses an economic model combined with several models used in the domain of market economy in order to resolve conflicts encountered between replicas. This service of consistency management proposed begins with a preliminary stage which consists in making a pre-processing before the service is started. This preliminary stage is composed of three phases (see Figure 2):

a. The phase of collection of information: to have a life on the status of the site, the manager replicas (MR) collects information on each data (metadata) periodically each node containing replicas of this data. This information can be represented the version of the vector of each replica, the timestamp and the origin of the update,…

b. The phase of information analysis: this step, which is based on the information collected to calculate a set of measures that we used allows, for example, the average standard deviation, distance, the number divergence average per site, the number conflict average per site, the distance a local site,…

c. The phase of decision: Starting metrics generated by the analysis phase, the management service consistency decides or not on the trigger the spread of updates, intra-or inter-site locations.

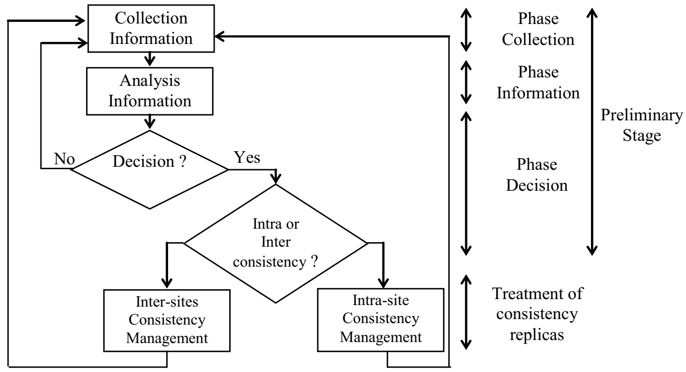

Fig. 2. Preliminary stage of consistency management in Data Grid

4.1 Intra-site consistency

The essential goal of consistency management in the level 0 is to make converging replicas of a data, stored in the same site, towards local reference replica. This service of consistency management, inspired of the optimistic approaches (Belalem & Slimani, 2007; Saito & Shapiro, 2005), supports the response time compared to coherence. On the basis of this objective, it is thus possible that the replicas of a file can diverge. Thus, a user can, at a given moment, observe copies of the same file with different values.

The intra-site process of consistency management is composed mainly of the following tasks:

Treatment of requests: This task has as a role to carry out a request subjected by client;

Propagation: Since we accept the fact that the replicas of a file can diverge, the task of propagation consists in making a transfer of the updates within a site given towards replicas which are not up to date. This propagation ensures, that at the end of a finished time, the replicas will converge towards the local reference replica;

Tolerance with the faults: The purpose of this task is to deal with the failures of nodes of site given and a site to the level of all the grid;

The Model Updating: This task fulfils the functions necessary to the model updating, and in particular the integration of new nodes in a site.

One of two main strategy replications can be chosen for each site to accommodate requests from users:

- Intra-site of consistency management with single-master strategy: The process of intra-site consistency management, with single-master strategy (Goel et al. 2005; Saito & Shapiro, 2005), starts by selecting a replica master among all the replicas stored in the nodes of a given site. The node containing this replica will be called the manager of replicas of a file given inside a site. In the single-master strategy, a request of a client of writing type, as for a given file, should be carried out only on this master, whereas a request of reading can reach any replica of this file.

- Intra-site of consistency management with multi-masters strategy: For intra-site of consistency management with multi-master strategy, a request of a client, type reading or writing, can be carried out on any node containing a replica of the file called upon by the request. When a client subjects a request $Req_i(k)$ from a site S_i using replication multi-masters strategy, then this request will be able to reach any replica p of the object O_k of a node of the site S_i.

We can summarize the various stages of sub-service intra-site consistency by the Algorithm Intra-Site Consistency:

Algorithm 1 Intra-Site Consistency
Begin / * Is a request arriving at the site controlled by MR_I */ **Switch** (Strategy) **Do** **Case** Single Master: **If** Master = free **Then** treatment of the request of the customer **Else** to deposit the request in the queue of the site **Case** Multi Master: **If** ∃Master = free **Then** treatment of the request of the customer **Else** to deposit the request in the queue of the site **EndSwitch** **If** Critical_situation(MR_i) **Then** Algorithm Inter-sites consistency /* $Site_i$ in Crisis */ **Else** Propagate of updates. **End**

4.2 Inter-sites consistency

The main objective of inter-sites consistency management, in the level 1, is to converge replicas of the data grid to a reference replica, using economic models of real market (Buyya & Vazhkudai, 2001).
This sub-service management of the consistency between sites using a pessimistic approach, it is composed of the following steps:

1) Selecting a representative: choose a representative of a site among all nodes, or the master or a master among masters of the rest according to a selection (for example, is the reply that suffered more updates, the most popular by a coefficient of reliability of the storage element, etc.).
2) Collection of information: One of the features of the representative is to receive information on the replicas (the meta-data) of the corresponding site through the management of replicas.
3) Collaboration: a representative collaborating with the other representatives of other sites, to converge the replicas of the same data.
4) Detection and resolution of conflicts replicas: If a conflict is detected, then the negotiation process for the resolution of conflict is triggered.

5. Process of conflict resolution

The collaborative relationship is based on the principle of domination replicas (Hasegawa et al. 1998; Ratner et al. 1997). If this dominance is not successful, the process detects conflicts; it is at this moment that the resolution process is started. A second situation that can trigger this process of resolution is that the analysis representing the state of its corresponding site through managers' replicas, if the replicas of the same data are too different from a threshold of tolerance, site is considered as being in crisis.

We say a site is in a critical situation if its versions replicas of the same object are too dispersed. In this case, it is called a site in crisis. The main stages of this consistency are represented in Figure 3:

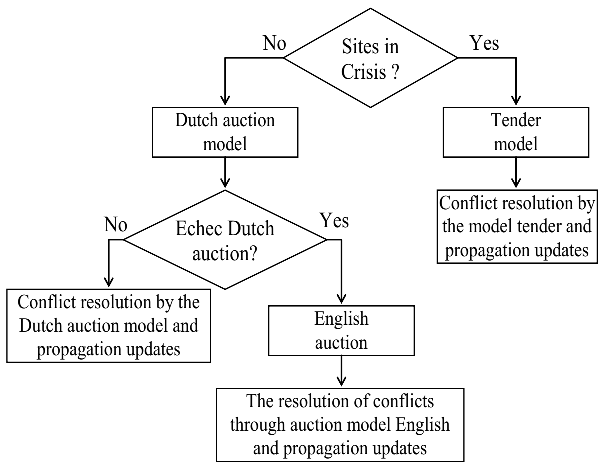

Fig. 3. The main stages of process of conflict resolution

If the trigger event is then checked, one of the two situations is encountered:

Are one or more sites are in a crisis situation, ie, that the manager (MR) of aftershocks following the analysis of information collected said that the information collected is very dispersed. Because of this situation that the decision to trigger the sub-service the overall consistency is taken. In this situation, an economy model type tender is better (see Algorithm Call_Tender).

_Algorithm 2 Call_Tender_

```
Calculate τi , Di, σi  /* measurements of i^th MR */
candidates ← false, Nbr_candidates ← 0
/* Nbr_candidates represents the sites candidates which can help
the site in crisis */
/* A site is called stable if is not in crisis */
for  all elements of the group of sites in crisis do
Representative_a ← MR of site in crisis
j ← 1
repeat
   Representative_b ← representative of the stable site
    if (|τa - τb| < ε) ∨ (|Da - Db| < ε) ∨ (|σa >σb| < ε) then
          Nbr_candidates ← Nbr_candidates+1
      end if      /* Where   ε << 1 */
      j ← j +1
 until  j ≥ Number of elements of the group of stable sites
 if  (Nbr_candidates > 1) then
   candidates ← true
   Algorithm Negotiation /* Negotiation of candidates */
 else
  Propagation of the updates of the stable site to the sites in crisis
 end if
end for
```

The result of this algorithm can be:
Several MR are candidates to offer their services to sites in crisis, then a negotiation (see Algorithm 3) is underway between them, to choose the site stable (non-crisis) to resolve the crisis sites to propagate its information;
If one MR selected by the algorithm tender, which will spread its information to the sites in crisis.

The negotiation process called by the algorithm of the invitation to tender allows to select the best stable candidate which solves the critical situation of a site in crisis

Algorithm 3 Negotiation

```
τi , Di, σi /* measurements of i^th MR for stable site*/
winner ← first of all sites stable  /* Candidate supposed winner */
τi , Di, σi  /* measurements of i^th MR */
/* Where MRi ≠  winner
no_candidate ← false
for  all MR – { winner } do
   if (τi  < τwinner) ∧  (Di < Dwinner)  ∧  (σi <σwinner) then
      winner ← reprensentativei
   end if
   winner propagates its updates with the sites in crisis ¶
end for
```

Either a chosen period in advance is reached and no site of the data grid is in crisis, then the economy model chosen was that of the Dutch auction (Algorithm 4).

Algorithm 4 Dutch_Auction

```
representative_max ← MR having the most recent version
version_reserves   /* the average of all vectors of versions */
τi , Di, σi  /* measurements of i^th MR */
no_candidate ← false
for all MR – {representative_max} do
    if (τmax > τi) ∨ (Dmax > Di) ∨ (σmax > σi) then
        if version_i ≤ version_reserves then
            no_candidate ← true
        else
            representative_max ← MR_i
            no_candidate ← false
        end if
    end if
end for
if (no_candidates = false ) then
representative_max propagates the updates with the whole of representatives
else
    Algorithm English Auction
end if
```

If no candidate is selected as the most favourable then we will make use of the economy model of English auctions (Algorithm 5).

Algorithm 5 ENGLISH AUCTION

```
representative_min ← MR having the oldest version
version_reserves   /* the average of all vectors of versions */
τi , Di, σi  /* measurements of i^th MR */
for all MR – {representative_min} do
    if (τmin > τi) ∨ (Dmin > Dlocal_i) ∨ (σmin > σi) then
            representative_min ← MR_i
    end if
end for
representative_min propagates the updates with the whole of the representatives
```

6. Impact of load balancing on the replicas

To improve the quality replicas, we have extended the service of consistency management proposed by a load balancing module (see Figure 4). The main objective of this module is that it allows to study and to measure the influence of the load balancing (Li & Lan, 2004) on quality replicas. We have defines two strategies of balancing that of requests and that of the replicas.

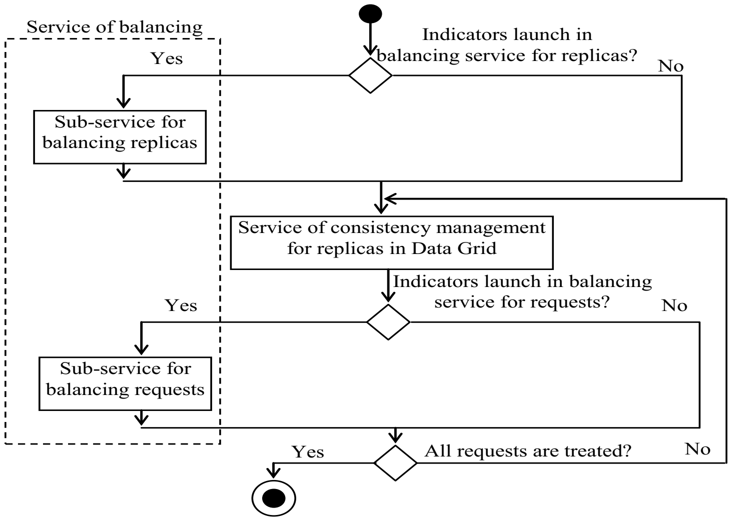

Fig. 4. Service of consistency management extended by the module balancing

6.1 Load balancing of replicas

In accordance the proposed model, we propose one load balancing level for replicas: Inter-sites. The indicators which we chose for sub-service of management of balancing of replicas are developed in Table 1 according to:

Measures	Definition
n_i^d	Number of replicas of the same data d in the $Site_i$
nbT_rep^d	Total of replicas of the same data d in Data Grid
$nb\_req_i^d$	Number of queries that access the same data d in the $Site_i$
nbT_req^d	Total number of queries that access the same data d in the Data Grid
ζ_i^d	The rate of requests accessing the same data d in $Site_i$ $= \frac{nb\_req_i^d}{nbT\_req^d} * 100$
$nb\_necessary\_rep_i^d$	Number of replicates necessary depending on the number of requests of the $Site_i$ $= \frac{\zeta_i^d * nbT\_rep^d}{100}$
ω_i^d	$\frac{n_i^d}{nb\_necessary\_rep_i^d}$
μ	The threshold of imbalance replicas

Table 1. Calculated measures in balancing of replicas

Design of sub-balancing service replicas is composed of three main steps:
1. *Phase of collection of information*: The manager starts by collecting following information:
- The number of replicas of the data *d* of its site;
- The full number of replicas of data *d* of Data Grid;
- The number of requests of the data *d* of its site;
- The total number of requests of data *d* in Data Grid.

2. *Phase of treatment*: The manager calculates:
-The rate of requests of the data *d* in its site;
- The necessary number of replicas according to the number of requests of the site;
- The report ω (See Table1).

3. *Phase Decision making*: During this stage, the manager of the site decides opportunity to start load balancing for replicas.

6.2 Balancing of requests

The management of the balancing requests service intra and inter-sites can be represented by the collaboration diagram as follows (Figure 5):

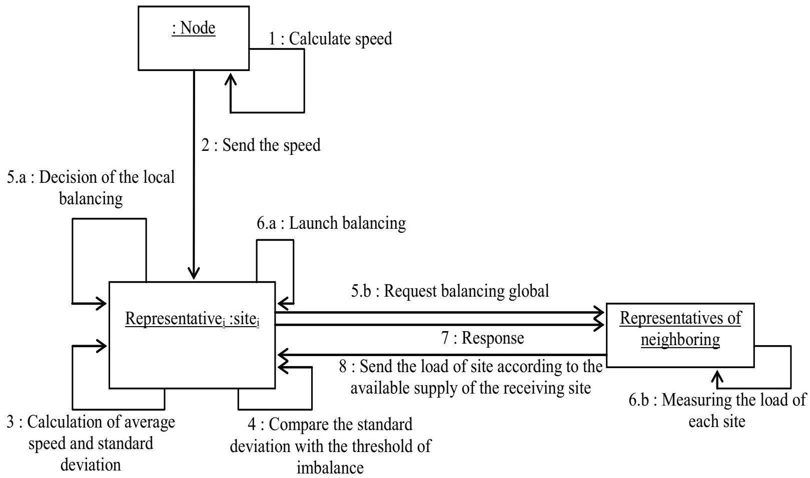

Fig. 5. Collaboration diagram

In accordance with the hierarchical structure of the proposed model, we distinguish two load balancing levels of requests: Intra-site and Inter-sites. The indicators which we chose for sub-service of management of balancing of requests are developed in Table 2.

Measures	Definition
m_i	The number of nodes in the $Site_i$
V_{ij}	The speed of $node_j$ owned $Site_i$ $= Nbr\_req\_processed_{ij} / Total\_nbr\_req_{ij}$
$\overline{V_i}$	The average speed of $Site_i$ $\overline{V_i} = \frac{1}{m_i}\sum_{j=1}^{m_i} V_{ij}$
σ_i	Standard deviation of the $Site_i$ $= \sqrt{[\frac{1}{m_i}\sum_{j=1}^{m_i}(V_{ij} - \overline{V_i})^2]}$
R_{size}	the size of the request
bdp	bandwidth intersites
TTij	transfer time of request of $Site_i$ to $Site_j$
$Demand_{ij}$	Number of requests to transfer from $node_j$ belonging to the $Site_i$
$Offer_{ij}$	Number of requests received by the $node_j$ belonging to $Site_i$
NR	List of recipients nodes
NS	List of sources nodes
$\overline{\omega}$	The threshold of hope
λ	The threshold of the imbalance

Table 2. Calculated measures in request balancing

The request "Demand" is the number of requests to transfer from one node to another node. On first calculates the difference between average speed of the $Site_i$ and speed of $node_j$ appointed (see equation 1),

$$dif_i = \overline{V_i} - V_{ij} \tag{1}$$

this difference is equal to the ratio between the number of queries are not processed by the total number of queries (equation 2)

$$dif_{ij} = \frac{Total\_nbr\_req_{ij} - Nbr\_req\_processed_{ij}}{Total\_nbr\_req_{ij}} = \frac{Nbr\_req\_not\_processed_{ij}}{Total\_nbr\_req_{ij}} \tag{2}$$

from Subtracting this demand, the number of complaints not dealt with, and we write (equation 3)

$$Nbr\_req\_not\_processed_{ij} = dif_{ij} * Total\_nbr\_req_{ij} \tag{3}$$

$$Demand_{ij} = Nbr\_req\_not\_processed_{ij} \tag{4}$$

The offer is the number of queries that the node can receive while remaining balanced. It first calculates the difference between the speed of node j and the average speed of sitei, this difference is called (equation 5),

$$\beta_{ik} = V_{ik} - \overline{V_i} = Nbr\_req\_not\_processed_{ik} / Total\_nbr\_req_{ik} \quad (5)$$

this difference represents the ratio between the number of queries are not addressed by the total number of queries (equation 5), from this formula it is deduced that the offer (equation 6) is equal to the number of queries untreated

$$Offer_{ik} = \beta_{ik} * Total\_nbr\_req_{ik} \quad (6)$$

This sub-service can run a balancing local (balancing intra-site), ie, within a site, if this balance can not be satisfactory, the sub-service triggered a balance between the sites of the grid (balancing inter-sites).

A. Balancing intra-site

In the case of balancing intra-site, three stages can be also used:

Estimate of the load of the site: This stage defines the mechanisms of measurement and of communication of load. Knowing the number of the nodes of the site like their respective capacities, each manager estimates the capacities of the site with which it is associated by carrying out the following actions: Considering the requests are more or less of the same size, we propose like index of load, the speed of treatment requests V. We consider the mean speed of the site starting from the information received periodically by its nodes. This measurement makes it possible to give us a sight on dispersion of speeds of the nodes. We chose to calculate the standard deviation with an aim of measuring the extent of the variations between the mean velocity of the site and speeds of its nodes. Each node sends the information of load to its associated manager.

Decision making: During this stage, the manager of the site decides opportunity to start a local balancing of load. For that, it carries out the following actions:

The state of load of the sites we can say that a site is in a state of balance when this variation is relatively weak. That means that the speed of each node converges towards the mean speed of its site.

State of balance: In practice it is a question of defining a threshold of balance, to leave we can say that the standard deviation tends towards zero and thus the site is in state of balance. Thus we can write: If ($\alpha < \lambda$) then the site is balance if not it site is in a state of imbalance.

Partitioning of the site: When a site is imbalance, we can consider it release of an operation of balancing of requests. To determine if a node of a site is in a suitable state to take part in a transfer of requests like source or like receiver, we divide the site in two groups of Nodes: (i) Overloaded nodes (sources), (ii) undercharged nodes (receivers). This classification depends on the difference between the speed of each node and that of its site.

Transfer of requests: To carry an operation of balancing of requests, we propose the following heuristics:

To calculate request (Demand), i.e., number of requests to be transferred necessary by overloaded node.

To calculate the offer, i.e., the number of requests to be received. We can distinguish three types of undercharged nodes:
The nodes which never received requests, and in this case Offer can have any number of requests
The nodes which treated all their requests Offer can have any many requests
The nodes which have treated requests and untreated requests In this case, if the offer is not able to satisfy the request sufficiently, it is not recommended to start a local balancing. To measure the offer compared to ask, If (Offer>Demand) then to start local balancing If not to start global balancing.

B. Balancing inter-sites
In the case of balancing inter-sites, three stages can be also used:
Estimate of the load of the site;
Decision making;
Transfer of the requests.
If a manager fails to balance his load locally, it estimates its charge compared to its vicinity. In the case of an imbalance, the manager decides to transfer his requests towards sites under charged closest (pertaining to its vicinity). In addition to the collection of information of load of his neighbours, the manager of the site must take account of the costs of communication induced by possible transfer of requests.

7. Experimental results and discussion

To study the management of consistency and its extended version of the module balance, compared to the conventional optimistic approach, we developed a simulator in Java to meet that goal. We used these simulations to a set of parameters. Table 3 describes the main parameters used in our series of experiments.

Parameters	**Interval**
Number of requests	[10..1000]
Number of data	[1..10]
Number of replicas per data	[10..100]
Number of nodes per site	[10..100]
Number of sites	[5..50]
Size of data (MB)	[5..100]

Table 3. Simulation parameters

In this work, we present three experiments: the first series of experiments used to study the impact of the number of requests on our proposed approach and the optimistic one, for the second series of experiments we measure the impact of the number of sites in the data grid on the results of both approaches.

7.1 Impact of the variation of number requests

In this series, the simulation results have been achieved with the following parameters: 5 sites, 30 nodes per site, 1 data, 30 replicas per site depending on the multi-masters, the bandwidth is set to 1000 Mb/ms for a star topology in intra-site and 500 Mb/ms inter-sites, we varied the number of requests (such as writing) from 25 to 150 in steps of 25.

A. Impact load balancing of replicas

Figures 6 and 7 show the behaviour of four approaches: ours called hybrid, hybrid with balancing, optimistic approach and the classical optimistic extended by balancing module. The curve is below the curve of the optimistic approach. We conclude that our proposed approach reduces the number of differences and conflicts in relation to the optimistic and that the extension of the approach by balancing module replicas to improve in a positive way the hybrid approach proposed.

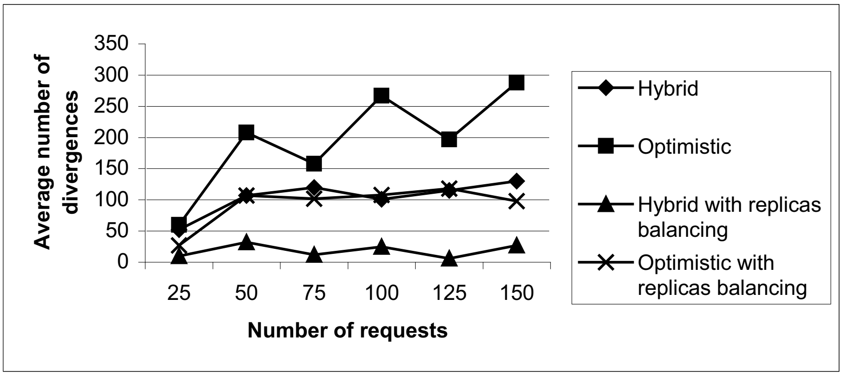

Fig. 6. Average number of divergences / Number of requests

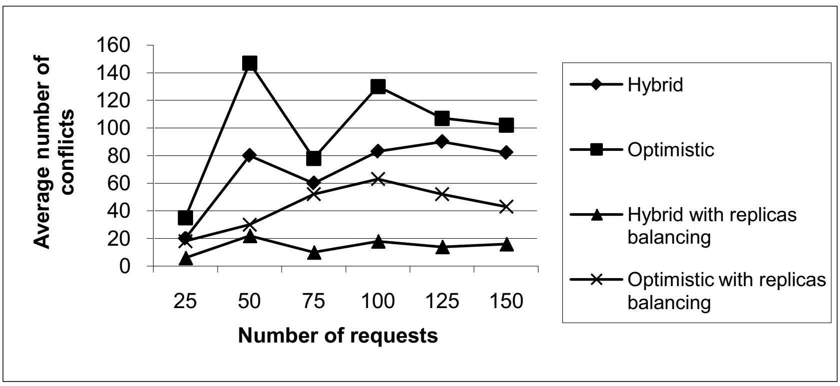

Fig. 7. Average number of conflicts / Number of requests

B. Impact load balancing of requests

Figures 8 and 9 show in a first time for the hybrid approach based on negotiation generates less divergences and conflicts over the traditional approach, and in a second time as the load balancing of requests sent customers by improving the quality of replicas of a very interesting way.

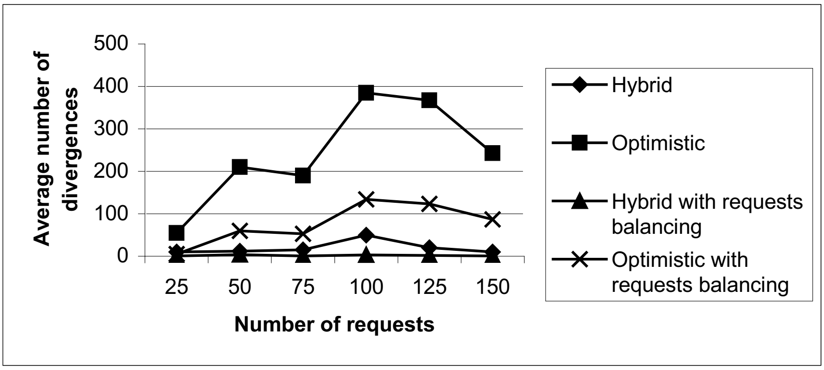

Fig. 8. Average number of divergences / Number of requests

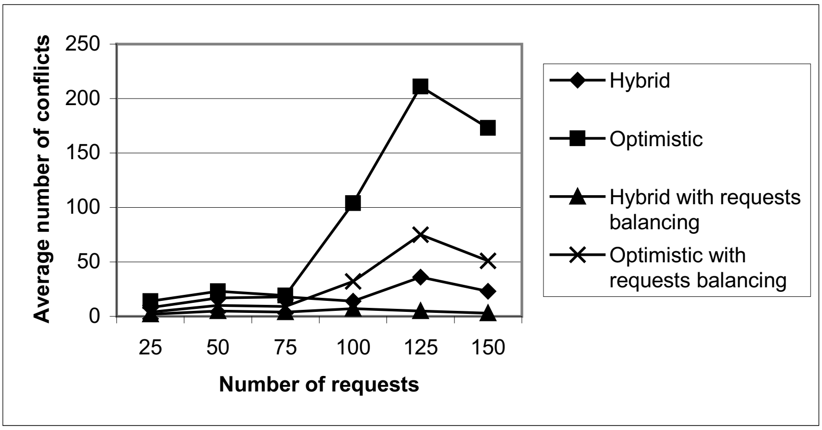

Fig. 9. Average number of conflicts / Number of requests

7.2 Impact of the variation of number sites

This series of simulations is the influence of the number of sites on divergences and conflicts. The parameters we used are: 30 nodes, 100 requests for records, data replicated 30 times in each site according to the multi-master, 1000 Mb/ms bandwidth for a star topology in intra-site and 500 Mb/ms inter-sites and we considered the number of sites ranging from 5 to 20 in steps of 5.

The main objective that we set for the following two simulations is shown how the approaches behave with the scalability of system.

A. Impact load balancing of requests

Figures 10 and 11 show, that the hybrid approach, is still better than the optimistic approach with balancing requests and without. We can also note that our proposal supports the scalability of the system.

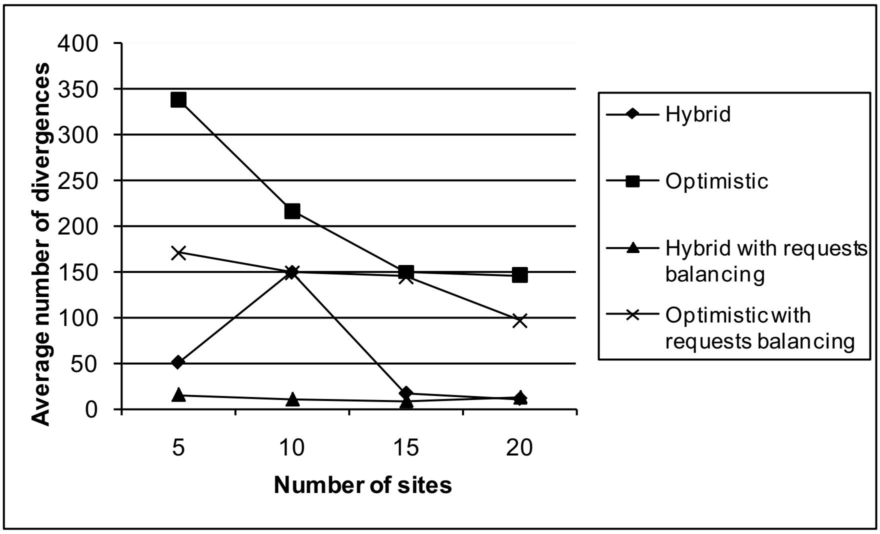

Fig. 10. Average number of divergences / Number of sites

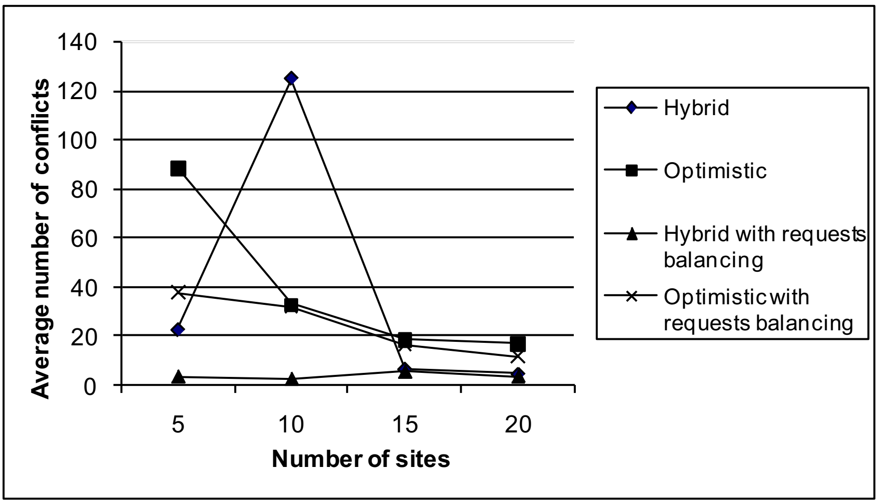

Fig. 11. Average number of conflicts / Number of sites

B. Impact load balancing of replicas

The same result is obtained by the figures 12 and 13. We note, from the number of sites 15, the hybrid approach proposed tends to the hybrid approach with by balancing the module (see Figures 12 and 13).

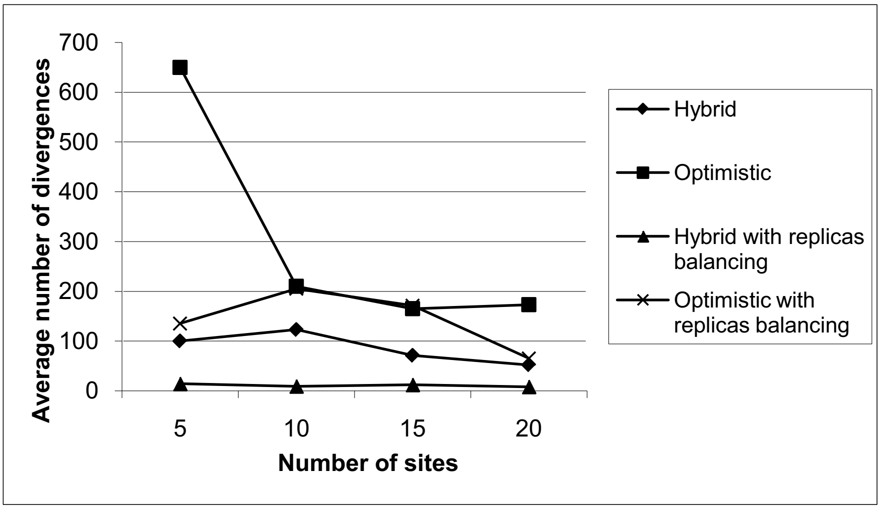

Fig. 12. Average number of divergences / Number of sites

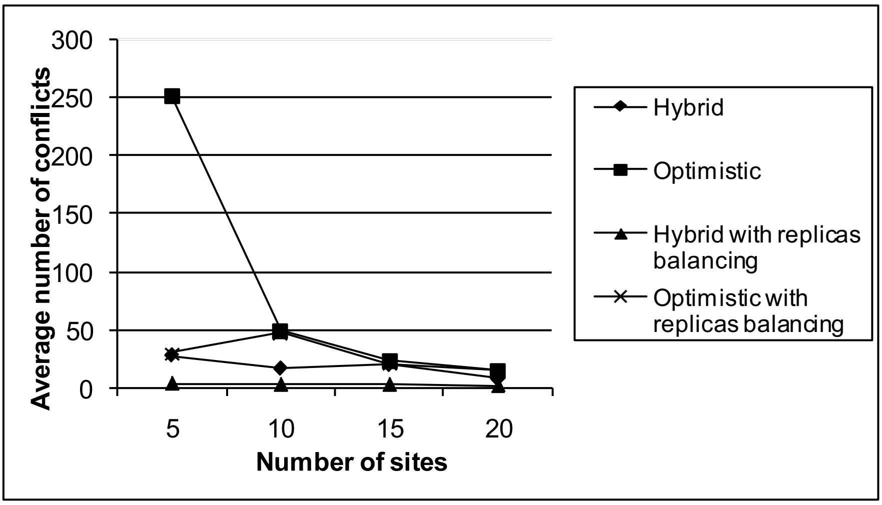

Fig. 13. Average number of conflicts / Number of sites

8. Conclusion

The replication technique is used for data management in distributed systems and grids to ensure data availability and fault tolerance. However, the use of the technique raises the problem of maintaining consistency of replicas. Unfortunately, to ensure the consistency of replicas, you must have a strong consistency, where the degradation of performance.

We have proposed a model for consistency management of replicas in data grids, which combines at the same time the pessimistic approaches, which support the quality of service (QoS), and the optimistic approaches which are focused on the improvement performance. Our effort in this contribution, aims to resolve conflicts between the replicas by using a collaborative negotiation between representatives of the nodes in the data grid. We try to show in this work that trading is influenced by the load balancing of the grid.
The results of the comparison showed that our approach generally ensures a better quality replicas while keeping the same system performance. From these results, we can conclude that the load balancing influence in a positive way on the quality of the replicas.
We want to show by this comparison that the factor balancing requests or replicas has a direct impact on reducing the number of divergences and conflicts between replicas. The work is far from over, we can suggest several perspectives that can be the subject of future work:
Implement our approach in the GLOBUS project;
Improve this approach to tolerate faults in the Data Grid;
Studying the behaviour of the proposed approach in Wireless Grids.

9. References

Belalem, G. & Bouhraoua, F. (2007). Dynamic Strategy of Placement of the replicas in Data Grid, *Malyshkin V. (Ed.) : Parallel Computing Technologies, 9th International Conference, PaCT 2007, Pereslavl-Zalessky, Russia, September 3-7, 2007, Proceedings, Lecture Notes in Computer Sciences (LNCS),* volume 4671/2007, pp. 496-506.

Belalem, G. & Slimani, Y. (2007). A hybrid approach to replica management in data grids, *International Journal Web and Grid Services (IJWGS). Vol.* 3, No. 1, pp. 2-18.

Buyya, R. & Vazhkudai, S. (2001). Compute Power Market: Towards a Market-Oriented Grid, *CCGRID'01, 1st International Symposium on Cluster Computing and the Grid*, May 15-18, 2001, Brisbane, Australia, pp. 574-581.

Cameron, D. G., Casey, J., Guy, L., Kunszt, P.Z., Lemaitre, S., McCance, G., Stockinger, H., Stockinger, K., Andronico, G., Bell, W. H., Ben-Akiva, I., Bosio, D., Chytracek, R., Domenici, A., Donno, F., Hoschek, W., Laure, E., Lucio, L., Millar, P., Salconi, L., Segal, B. & Silander, M. (2004), Replica Management in the European DataGrid Project, *Journal Grid Computing*, Vol. 2, No. 4, pp. 341-351.

Dorcey T. (1995). Cu-SeeMe desktop videoconferencing software. *Connexions,* Vol. 9, No. 3, pp. 42-45.

Foster, I. & Kesselmann, C. (2004). *The Grid 2: Blueprint for a new computing infrastructure*. Elsevier Series in Grid Computing. Morgan Kaufmann Publishers.

Goel, S., Sharda, H., & Taniar, D. (2005). Replica synchronisation in grid databases. *International Journal Web and Grid Services (IJWGS)*, Vol. 1, No. 1, pp. 87-112.

Gray, J., Helland, P., Neil, P. O., & Shasha, D. (1996). The dangers of replication and a solution. *In ACM SIGMOD International Conference on Management of Data*, pp. 173-182, Montreal, Quebec, Canada, 4-5 June 1996. ACM Press.

Hasegawa, K., Higaki, H. & Takizawa, M., (1998). Object Replication Using Version Vector, *In Proceedings International Conference on Parallel and Distributed Systems* (ICPDS'98), 14-16 Dec 1998, Tainan, Taiwan, pp. 147-154.

Kermarrec, A-M., Rowstron, A., Shapiro, M. & Druschel, P. (2001). The IceCube approach to the reconciliation of divergent replicas. *PODC '01: Proceedings of the twentieth annual ACM symposium on Principles of distributed computing*, pp. 210-218, Newport, Rhode Island, United States.

Kistler, J. J. & Satyanarayanan, M. (1992). Disconnected operation in the coda file. *ACM Trans. on Computer Systems*, Vol. 10, No. 1, pp. 3-25.

Li, Y. & Lan, Z. (2004). A survey of load balancing in grid computing. *High Performance Computing and Algorithms*, Computational and Information Science, First International Symposium, CIS 2004, Shanghai, China, December 16-18, 2004, Lecture Notes in Computer Science (LNCS), volume 3314/2004, pp. 280-285.

Molli, P., Oster, G., Rusinowitch, M. & Imine, A. (2002). Development of Transformation Functions Assisted by Theorem Prover. *The Fourth International Workshop on Collaborative Editing*, ACM CSCW 2002, New Orleans, Louisiana, USA.

Olston, C. & Widom, J. (2005). Efficient monitoring and querying of distributed, dynamic data via approximate replication. *IEEE Data Eng. Bull.*, Vol. 28, No. 1, pp.11-18.

Petersen, K., Spreitzer, M., Terr,y D. & Theimer, M. (1996). Bayou: replicated database services for world-wide applications, *EW 7: Proceedings of the 7th workshop on ACM SIGOPS European workshop*, pp. 275-280, Connemara, Ireland.

Ratner, D., Reiher, P. & Popek, G. (1997). Dynamic version vector maintenance. *Technical Report CSD-970022*, UCLA, June 1997.

Saito, Y. & Shapiro, M. (2005). Optimistic replication. *ACM Computing Surveys*, Vol. 37, No. 1, pp. 42-81.

Vidot, N., Cart, M., Ferrié, J. & Suleiman, M. (2000). Copies convergence in a distributed real-time collaborative environment, *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pp. 171-180, Philadelphia, Pennsylvania, USA.

Xu, J., Li, B. & Li, D. (2002). Placement problems for transparent data replication proxy services. *In IEEE Journal on Selected areas in Communications*, Vol. 20, No. 7, pp. 1383-1398.

Multimedia

Edited by Kazuki Nishi

ISBN 978-953-7619-87-9
Hard cover, 452 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

Multimedia technology will play a dominant role during the 21st century and beyond, continuously changing the world. It has been embedded in every electronic system: PC, TV, audio, mobile phone, internet application, medical electronics, traffic control, building management, financial trading, plant monitoring and other various man-machine interfaces. It improves the user satisfaction and the operational safety. It can be said that no electronic systems will be possible without multimedia technology. The aim of the book is to present the state-of-the-art research, development, and implementations of multimedia systems, technologies, and applications. All chapters represent contributions from the top researchers in this field and will serve as a valuable tool for professionals in this interdisciplinary field.