DRP-DRP: Data Replication Protocol for Disaster Recovery Planning

Hanan S. Al-Ghamdi
King Abdul Aziz University
hsaalghamdi@kau.edu.sa

Dr. Arwa Y. Al-Aama
King Abdul Aziz University
aalaama@kau.edu.sa

Abstract

In this paper, a new Data Replication Protocol for Disaster Recovery Planning (DRP-DRP) is proposed. The primary objective for developing DRP-DRP was to overcome the difficulties associated with the existing data replication protocols, namely synchronous and asynchronous protocols. The asynchronous protocol suffers from high replication latency and inconsistency of the replicated data, whereas the synchronous protocol suffers from high application latency and low throughput of business applications. Furthermore, none of these two approaches considers the criticality of the replicated data or business policy that should be achieved during the replication processes. Agent technology was used as an aid to develop DRP-DRP in order to minimize the required bandwidth cost for data replication and to provide an estimation of the network status. The correctness of the DRP-DRP protocol and algorithms were verified and the performance of the DRP-DRP was evaluated

1. Introduction

Nowadays, the concept of disaster recovery planning, or DRP, is a key concern and basic requirement for every business. Data replication is one of the major processes of the DRP. In general, there are two replication approaches, synchronous and asynchronous replication. Each has its benefits and problems [1]. Synchronous replication ensures that a perfect copy of the data at the primary site is maintained at a secondary site. With this replication method, every update request at the primary site must be processed and acknowledged from the remote site before the next update is allowed to occur [1]. This works well for replication within a local area network (LAN); however, extending this approach to transfer data over WAN results in significant latency problems of the business applications [2]. This latency is known as application latency [1]. In addition, synchronous replication incurs high bandwidth cost, and causes a dramatic degradation in the throughput and performance of the critical business applications. This can have a highly negative effect on business operations; thus, synchronous replication is limited to distances of 100 miles or less [1].

In asynchronous replication, every update request is acknowledged locally and, in parallel, immediately added to a queue of update requests waiting to be transmitted and stored at the remote site [1]. Therefore, asynchronous replication eliminates the application latency problems associated with synchronous replication. However, with asynchronous replication, the copy at the secondary site is not perfect. Furthermore, there is a time delay from the time that a change is made to the primary site and the time that the change is applied to the secondary site. This time delay is known as replication latency and is dependent on the network characteristics such as bandwidth, latency and packet loss.

Therefore, there is a need for a better data replication approach that can combine the best features of each of the existing approaches while avoiding their disadvantages. The new data replication approach should have the ability to maintain data consistency over a WAN connection with minimum replication and application latency and with a low bandwidth cost.

Moreover, the new approach should also have the ability to satisfy the business policies of the disaster recovery plan and take into account the criticality of the business applications which is defined in term of Recovery Point Objective (RPO) of the business application. Finally, it is also required that the new approach have the ability to react and adapt to the network dynamic changes.

This paper proposes a new approach for data replication, called DRP-DRP (Data Replication Protocol for Disaster Recovery Planning) which uses agents as an aid to satisfy requirements associated with DRP. Agents are special software components that have the ability to perform tasks on behalf of users or other programs autonomously [3]. An agent is *autonomous*, in that it has the ability to work without humans or other agent intervention and has control over its actions and states. It is also *social*, in that it cooperates with humans or other agents to achieve its tasks. An agent is also *reactive* or *adaptive*, in that it has the ability to monitor its environment and respond dynamically to changes that happen in the environment [4]. Moreover, an agent can be *mobile*, with the ability to travel from host to host in the computer network and perform its tasks at remote locations [5].

The proposed DRP-DRP protocol was implemented as a multi-agent system where multiple mobile agents were used to perform the replication task.

The rest of this paper is organized as follows. Section 2 describes the DRP-DRP multi-agent protocol, including its objectives, its operations, and its algorithms. Section 3 describes the performance evaluation test. Section 4 presents and discusses the results of the performance test. Section 5 concludes the paper.

2. DRP-DRP: Data Replication Protocol for Disaster Recovery Planning

2.1 DRP-DRP Overview

The proposed protocol considers the data replication between two sites: the primary server and the secondary server. The following assumptions are made:

- The two servers are connected over a WAN and have the same initial settings of their internal databases.
- The protocol considers only update requests. Read requests are not considered since there is no need to replicate data in this case.
- The primary server receives update requests from the clients running different business applications with different RPO needs.

The DRP-DRP basically prioritises the replication process based on the RPO of the replicated data and the available bandwidth to ensure the validity and the availability of the most critical data to the business on the secondary site. The protocol proposes maintaining data consistency by sending several mobile agents with several update requests to minimize the bandwidth cost associated with synchronous and asynchronous protocols. The mobile agents are used to group and replicate data items based on their RPO values. However, the WAN nature is dynamic; therefore, the decision about the data to be replicated should be taken dynamically. This need leads to the usefulness of using agents as they have the ability to sense and monitor environmental changes.

2.2 DRP-DRP system model

DRP-DRP consists of three main agents: *ReplicationMasterAgent*, multiple *MobileAgents* and *EstimatorAgent*. Figure 1 illustrates the general system architecture.

The *ReplicationMasterAgent* controls the replication process and performs the prioritization process. The *MobileAgent* carries the updates to the secondary server. The *EstimatorAgent* gives the multi-agent system estimations about the network status.

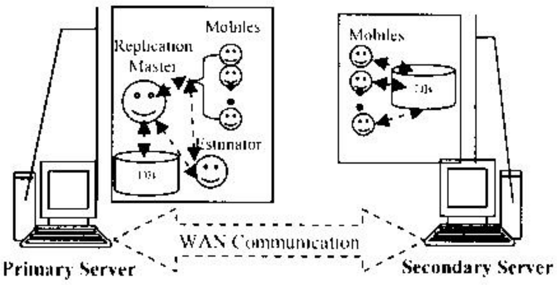

Figure 1: DRP-DRP system architecture

2.3 Data structures

The *ReplicationMasterAgent* maintains the following data structures:

- RPO List (*RPOList*): a list of RPO values of all business applications.
- Ready to Travel List (*RTL*): a list of the mobile agents' IDs. Each mobile agent is given a unique ID in the list.
- *EstimatedTime*: a long value, measured in milliseconds, of the estimated travel time between the primary site and the secondary site.

Each *MobileAgent* in the system has two main data structures:

- *TimeToTravel*: is the time in milliseconds that tells the mobile agent when it should move to the secondary site.
- *UpdateRequestsList* is a list of update requests carried out by the mobile agent, implemented as a Sorted Set java object. The sorting is based on the RPO.

2.4 Algorithms

The *ReplicationMasterAgent* implements two main behaviours running in parallel: *UpdateDataBase* and the *RTLManager*. The *UpdateDataBase* behaviour updates the data item and sends a 'COMMIT' message to the application immediately. At the same time, the

RTLManager receives the incoming update request and behaves according to the RPO of the application that sends the request. Then the *RTLManager* decides either to create a new mobile agent to carry the request or to add the request to the *UpdateRequestsList* of a mobile agent already existing in the *RTL* list. The *RTLManager* is implemented as an internal behaviour of the *ReplicationMasterAgent*, and not as a standalone agent, to reduce the amount of communications and message passing in the system.

Algorithm1 illustrates the operations performed by the *ReplicationMasterAgent* and Algorithm 2 illustrates the operations performed by the *MobileAgent*.

However, the main concern to be discussed here is to show how these algorithms help to satisfy data replication requirements for disaster recovery planning.

First, we have to show that in the case of a disaster, data loss at anytime will be acceptable by the business. Second, we have to show that in the case of low bandwidth availability, DRP-DRP does not increase either the application latency or the replication latency and does not affect the application's throughput.

Indeed, RTL can be considered as a waiting list of data to be replicated. However, the data waiting time depends on the RPO value, which is a business policy, and the estimated travel time. Therefore, in the event of a disaster, all data loss should be the data with a large RPO; and loosing this data at the time of a disaster is acceptable to the business. Furthermore, in DRP-DRP, updates are carried out by multiple mobile agents. After being dispatched, the mobile agents become independent of the dispatcher and can operate autonomously [5]. Therefore, in the case of a disaster, all data that must be replicated before the time of the disaster should be either already replicated or on the way to be replicated.

Algorithm 1: Operations performed by the Replication Master Agent

```
Initialization:
Initialize the RPOList
Get the EstimatedTime
WaitForUpdate:
IF 'UPDATE ' message is received
   THEN
          Update internal DB
Send 'COMMIT' message to the requester application
Check the RPO of the request
          Check the RTL of the mobile agents
IF there is a mobile agent in the RTL that satisfies the condition
(TimeToTravel < (RPO + RequestTime))
          THEN
                Add DataItem of the 'UPDATE' message to
       UpdateRequestsList to the mobile agent which
                has the maximum TimeToTravel satisfying the
condition
          ELSE
                Create a new mobile agent with
TimeToTravel := RequestTime + RPO – EstimatedTime;
                Add DataItem of the 'UPDATE' message to
       UpdateRequestsList
                Put the MobileAgentID at the top of the
RTL
Upon Receipt of 'ESTIMATE' message from Estimator Agent
          Update EstimatedTime
Upon Receipt of 'REMOVE' message from Mobile Agent
Remove the MobileAgentID from the RTL
SendMoveRequest:
Send 'MOVE' message to the mobile agent which resides on the
top of the RTL
Upon Receipt of 'PERMISSION' message from Mobile Agent
Check the RTL
Send MOVE message in the absence of higher priority Mobile
Agent
```

Algorithm 2: Operations performed by the Mobile Agent

```
Wait in the RTL until CurrentTime = TimeToTravel
 Upon Receipt of 'MOVE' message OR CurrentTime –
TimeToTravel
    Send 'REMOVE' message to the
    ReplicationMasterAgent
    Move to the secondary site
Upon Receipt of 'UPDATE'
                Update the UpdateRequestsList
Upon arrival at the secondary site:
Update all data items in the UpdateRequestsList
```

However, showing that DRP-DRP has a good performance in the case of low bandwidth availability can be twofold: first, upon the receipt of an update request, DRP-DRP updates the internal database on the primary site and sends the update request to the appropriate mobile agent in parallel. Then the DRP-DRP sends a COMMIT message to the application immediately. Therefore, in the case of a poor WAN connection, the application latency should not be increased by the DRP-DRP. Second, in the DRP-DRP, critical data has higher priority in the replication process and is replicated very frequently. However, to ensure that the replication latency of non-critical data is not increased, in the absence of critical data, each mobile agent, at regular time intervals, sends a *PermissionRequest* to the *ReplicationMasterAgent* which in turn decides if this mobile agent should move or should stay longer in the *RTL*. The time intervals are determined by each mobile agent based on its initial *TimeToTravel*. So, each mobile agent has different intervals. The *ReplicationMasterAgent* uses its internal knowledge of the *RTL* to decide if the requester agent can move or not. Then it replies with a 'MOVE' message to the agent which can move. The *ReplicationMasterAgent* also checks the *RTL* at regular intervals equal to the least RPO value defined. If there is only one mobile in the system it sends a 'MOVE' message to that agent. The mobile agents are created by the *RTLManager* behaviour of the *ReplicationMasterAgent* and waits in the *RTL* until the Time to Travel expires or a request to 'MOVE' is received from the *ReplicationMasterAgent*. While waiting in the *RTL*, the incoming update request messages are added to the *UpdateRequestsList*. Before moving to the secondary site, the mobile agent sends a 'REMOVE' message to the *ReplicationMasterAgent* to remove it from the *RTL*. Upon arrival to the secondary site, the mobile agent updates all data items locally.

3. Performance evaluation

To demonstrate the effectiveness of the proposed DRP-DRP, three simulators were built to simulate the DRP-DRP, synchronous and asynchronous approaches. All implementations were built using the Java programming language and the JADE agent framework version 3.5. An interface was developed for each implementation to set up the experiments and to clear the results of the previous experiment. A *RequestGeneratorAgent* was used to simulate business applications. For each application, a database, using MySQL, was created. The experiments were conducted using two running machines connected by a WAN simulated-connection. Shunra VE desktop was used to simulate eight different WAN connections with different bandwidth, latency and packet loss values. Bandwidth values range from 256 kbps to 1024 kbps; and for each bandwidth, two values of latency and packet loss were selected. However, in each test scenario, six application agents were generated to simulate SCADA and front-office systems. The RPO values of these systems were selected from the range of applications' RPO presented in [6]. After each experiment, five metrics were calculated: data consistency, business policy satisfaction, throughput, average of application latency, and replication latency.

4. Results and discussion

The results presented in Figure 2 shows that the DRP-DRP maintains 100% data consistency in all test scenarios. This is due to the fact that DRP-DRP transfers data based on criticality level of the data, and the estimated transfer time. In addition, if there is any change in the network characteristics, the DRP-DRP will adapt to that change by updating both, mobile agents and the *ReplicationMasterAgent*.

The business policy satisfaction results, presented in Figure 3, show that DRP-DRP achieved 100% business policy satisfaction in most scenarios. Indeed, the worst results of DRP-DRP were 92% and 84%. However, theses results were in the first two test scenarios where the network conditions were the worst with 256 kbps bandwidth. Therefore, to achieve 100% business policy satisfaction, DRP-DRP should be used over a WAN connection with 512 kbps bandwidth at minimum.

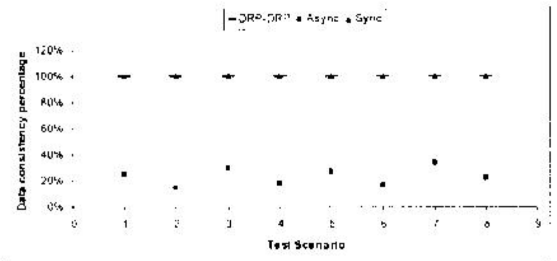

Figure 2: Data consistency

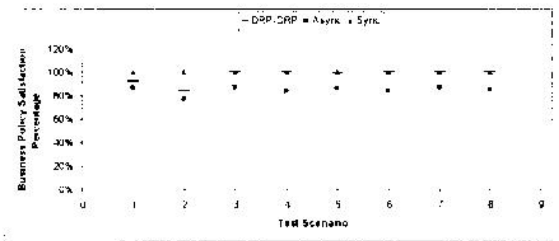

Figure 3: Business policy satisfaction

Figure 4 shows that the application latency of DRP-DRP is close to the asynchronous protocol. This is due to the fact that DRP-DRP, in parallel, updates the data item on the primary site and sends a 'COMMIT' message to the application immediately. At the same time, the data item is sent to the *RTLManager* which adds the data to the *RTL* list. In our opinion, the cause of the small difference between DRP-DRP and the asynchronous protocols is that DRP-DRP implements additional behaviours to process the incoming update request which in turn introduce additional overhead.

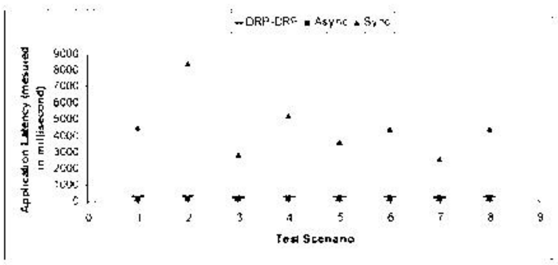

Figure 4: Average application latency

Figure 5 shows that the throughput of asynchronous and DRP-DRP were closer to each other. This is because DRP-DRP responds immediately to the application after updating the primary site. Moreover, this feature of the DRP-DRP makes it unaffected by the network characteristics. Therefore, in the case of dynamic network changes, DRP-DRP is highly recommended.

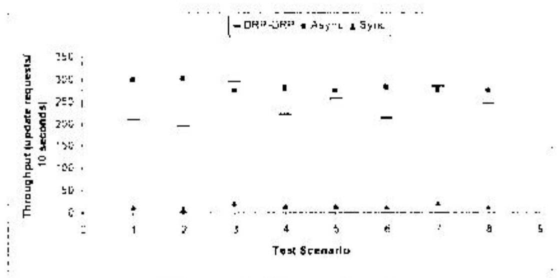

Figure 5: Throughput

Figure 6 shows that DRP-DRP results in higher replication latency than synchronous replication but much less than asynchronous replication. It is worth noticing that although the DRP-DRP maintains a waiting queue. *RTL* and the waiting time for non critical data, is supposed to increase the total replication latency. Howebver, the total replication latency was not increased by DRP-DRP. This is because in DRP-DRP each mobile agent, at regular time intervals, sends a *PermissionRequest* to the *ReplicationMasterAgent*, which in turn decides if the requester agent can move or not. It replies with a 'MOVE' message to the agent which can move. The *ReplicationMasterAgent* also checks the *RTL* at regular intervals equal to the least RPO value defined. If there is only one mobile in the system it sends a 'MOVE' message to that agent.

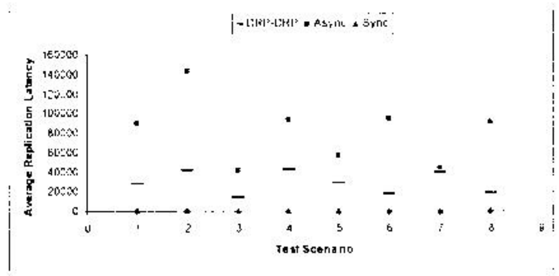

Figure 6: Average replication latency

5. Conclusion

In this research, we proposed a new Data Replication Protocol for Disaster Recovery Planning (DRP-DRP) to overcome the difficulties attached to the existing data replication protocols. Agent technology was used as an aid to develop DRP-DRP. The correctness of the DRP-DRP protocol and algorithms were verified and the performance of DRP-DRP was evaluated against synchronous and asynchronous approaches. Three simulators were developed for evaluation purposes. The performance test was conducted over a simulated wide area network. The results showed that DRP-DRP has improved performance over a WAN. Among its feature, it may be concluded that DRP-DRP:

- has the ability to maintain 100% data consistency over a WAN connection without dramatic degradation in business application throughput or a high increase in either replication latency or application latency.
- has the ability to satisfy business policies of the disaster recovery plan.
- has the ability to adapt to the network dynamic changes.

It is also worth noticing that DRP-DRP can be considered as a generic method and can be applied to implement different kinds of applications wherever there is a priority among the data or processes in a distributed environment.

A number of aspects that can be further researched include methods for maintaining data consistency among multiple sites, choosing the appropriate size for a mobile agent, replicating unstructured data such as files, and having only a partial storage on the secondary site.

6. References

[1] Info-Tech Research Group, "Building a Comprehensive Disaster Recovery Plan". Info-Tech Research Group, September 2003.

[2] Christoph Mitasch, "Server-Based Wide Area Data Replication for Disaster Recovery", June 2004, available at: http://www.linux-ha.org/_cache/BusinessContinuity__da.pdf. (Visiting date: March 2008)

[3] Kozma, J.; "Intelligent agents " Potentials, IEEE Volume 17, Issue 2, Apr-May 1998 Pages.16 – 19.

[4] Fabio Luigi Bellifemine, Giovanni Caire and Dominic Greenwood, "Developing multi-Agent systems with JADE", Willey, 2007

[5] Horvat, D., Cvetkovic, D.; Milutinovic, V., Kocovic, P., Kovacevic, V.; "Mobile agents and Java mobile agents toolkits " System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on Jan 4-7 2000 Page: 10.

[6] Cegiela, R., "Selecting Technology for Disaster Recovery" Dependability of Computer Systems, 2006. DepCos-RELCOMEX apos.06. International onference on Volume , Issue , May 2006, Pages:160 – 167