Data synchronization in a replicated distributed database

Xiaomei Zheng

University of Tampere
Department of Computer Sciences
Computer Science / Software Development
M.Sc. thesis
February 2007

University of Tampere
Department of Computer Sciences /Computer Science / Software Development
Xiaomei Zheng: Data synchronization in a replicated distributed database
M.Sc. thesis, 51 pages, 2 appendix pages
February 2007

Abstract

This study analyzes data protection and disaster recovery technologies and existing solutions. This work is carried out in Charging and Service Control business group inside Nokia Network. Our intelligent network has high availability implemented already in one data center by Oracle RAC. In order to achieve the continuously data availability in the event of a site disaster, another data center at a remote site is introduced into our IT infrastructure. The target is to minimize the downtime associated with the outage and preventing data loss in a site disaster, as well as to derive the most out of the disaster recovery infrastructure even in times when there is no disaster.

Two data centers mean two locations for data. The focus of this study is to find out how the data is synchronized between two databases residing on two different sites. The main goals are to understand what the efforts and costs are to adopt a certain solution, what the performance is, what problems can occur, how they can be solved and to illustrate the inherent limitations and challenges of some technologies.

The study deals with the general site disaster tolerance requirements from our customers. The problems found concern, for example, replication conflicts, limitation of inter-site connection technologies, essential of synchronous and asynchronous mode. The problems are analyzed on the basis of literature in the field of data synchronization in high availability and disaster tolerance environment.

Based on the comparison and balancing of costs, performance and availability, a compromised cost-efficient solution is proposed for our distributed database environment which has the essential of replication conflict. This solution is based on standby database technology that keeps a standby copy of the database at a remote site synchronized with the primary site.

Key words and terms: synchronization, replication, disaster tolerance, high availability, synchronous and asynchronous mode.

Contents

Acknowledgement

This thesis is made for the Charging and Service Control Product Line in Nokia Networks which I would like to thank for the possibility of writing the thesis.

First and foremost, I like to express my sincere thanks to my supervisors Prof. Jyrki Nummenmaa for his guidance and education during my studies in Computer Sciences at the University of Tampere. I am deeply grateful to his patience in reading and revising my thesis. In addition, I would like to thank Prof. Erkki Mäkinen for his advice and instructions. I also appreciate Virginia Mattila for her proofreading and checking the linguistic form of this thesis.

I would also like to thank my colleague Johanna Valimaki and Kari Niemi for their fruitful ideas which enlightened me in this thesis. Johanna also gave me her research document as reference. Special thanks to M.Sc. Atte Leppanen for giving me this chance to carry out this work as well as the inspiring working atmosphere inside service logic group. Also I am very grateful to my instructor M.Sc. Jukka Ahonen who had helped me to find the topic for this thesis.

I wish to thank my parents for their understanding and encouragement as well as my husband Shengfan Hou for his support.

Finally thanks to all friends who have stood by and shared the good and bad days.

Tampere, February, 2007

Xiaomei Zheng

1 Introduction

1.1 Background

System downtime is expensive or maybe very expensive depending on the business. The discussion in this thesis is focus on telecommunication business which is using Intelligent Network. The Intelligent Network (IN) is a network architecture for both fixed and mobile telecommunication networks. It allows operators to differentiate themselves by providing value-added services in addition to the standard telecom services [Wikipedia]. In an intelligent network environment, whatever critical system or application fails or is taken offline, the cost can run at the level of millions of dollars per hour or even more because our customers are mainly the telecommunication operators all over the world. A system failure means a call can not be connected and the telecommunication will be interrupted. We can directly calculate the cost of lost sales or transactions, but the damage to the customer relationships and the future health of the business are uncountable. The longer the downtime lasts, the more damage is done. We will lose the competitive advantage if our customers (operators) lose their customers (subscribers) because they adapt our network solution and use our applications. Telecommunication is such an industry that it needs the service to be available 24 hours a day, 7 days a week. All the underlying systems, applications and resources which might cause the interruption of communications are critical components in intelligent network. Considering that so many possibilities could be a source of system downtime, e.g. accidents, equipment failure, human errors in management and natural disasters, we understand that ensuring the high availability (HA) is costly and complex. A rough rule used to be that making an application highly available could triple the cost of deploying [Oracle, 2004].

1.2 Research problem

HA is a wide topic and it is implemented by many technologies in data backup and recovery, data replications, system monitoring and so on. Our customer (operator) requires us to introduce a disaster tolerance feature into our HA network product. They want to have two geographically distant sites which handle traffic at the same time. These two sites are connected to each other. The database on one site is synchronized with the other one, forming a robust distributed database environment. When disaster happens at one site, the transaction monitor routes transactions to the other site. This remaining working site continues call processing.

This requirement motivates the research problem of database synchronization. How to synchronize data to a remote site in a replicated distributed database system will be our focus in this thesis.

The main objective of the study is to solve the research problem by analyzing the individual technologies used in remote data replication/synchronization domain and by identifying crucial limitations in each type of solutions. A secondary objective of the study is to investigate whether the existing solution can be directly used to solve replication conflicts without any customizations.

The study is based on a literature survey and solution evaluation. Typically, there are certain limitations in a solution which restricts its potential use. The parameters in the requirements specify the conditions under which a solution is applicable and feasible. The advantages and disadvantages of technologies inside each solution are analyzed and compared. The comparison is based on performance and cost data from the literature.

The thesis comprises six chapters. The objectives in developing a remote replication solution are presented in Chapter 2. The characteristics of our network application are presented in Chapter 3. The customer requirements are analyzed in Chapter 4. In Chapter 5, the existing replicating and synchronization technologies are presented with their inherent limitations, the solutions are categorized by different parameters, and an integrated solution is introduced. Conclusions are drawn in Chapter 6.

2 Objectives in developing a remote replication solution

The technologies in implementing a remote replication solution are complex. Making the decision on which technology to choose requires an analysis of the following competing objectives [HP, 2006]:

• High availability: Does our business need continuous access to data without downtime?

• Disaster tolerance: Does our business need data to survive a site disaster?

• Distance and Performance: What is the effect of distance on replication throughput?

• Cost: How expensive are the data transmission lines between two sites?

The rest of this chapter examines these objectives in detail.

2.1 High availability

High availability reduces the risk of downtime through redundant systems, software, and IT processes with no single point of failure (NSPOF) [Weygant, 2001]. Providing redundant data is the key point in contributing to high availability.

The recovery time objective (RTO) is a measure of high availability. It is the length of time the business can afford to spend returning an application to operation. It includes the time required to detect a failure, to rearrange the data access, and to restart the application on a new server. RTO is usually measured in minutes or hours, and, occasionally, in days. [HP, 2006] The target RTO in our system is suggested to be from 15 minutes to one hour. A shorter RTO increases the need for products that automatically fail over applications and data.

If only high availability is needed, but not disaster tolerance, local and remote sites can be in the same room, building, or city. Distance and its effect on cost and performance are not important issues in this case. The database high availability feature has already been implemented by Oracle RAC configuration in our network [Ahonen, 2004a]. What we need to dig out in this thesis is how to strengthen the disaster tolerance capability.

2.2 Disaster tolerance

Disaster tolerance uses redundant technology to enable the continued operation of critical applications during a site disaster. There can be many redundant sites in practice. In order to simplify the solution and limit the research range, we constrain our solution to building two separated sites. One is regarded as the redundant site to the other. If these two sites are separated by a distance greater than the potential size and scope of a disaster, each site is protected from a disaster on or near the other site. Such a solution requires building two copies of application data at sites that are far enough apart to provide disaster tolerance.

2.3 Distance and performance

The location of the remote site is critical in disaster tolerance solution. The size of the threat to each site determines the required distance between local and remote sites. It varies from a few kilometers to intercontinental distances depending on different customer needs.

Most of the replication software can move data at extreme distances. However, the speed of light in fiber optic cables (5 microseconds per kilometer) causes inherent delays, called

latency. At extreme distances, latency is the limiting factor in replication performance, regardless of bandwidth [HP, 2006].

The greater the distance, the greater the impact inter-site latency has on replication performance. HP [2006] presents two ways to determine inter-site latency. They can be used to estimate the latency when we have detailed requirement specification from customer describing their network environment:

• Network utilities: For an existing network, we use network utilities such as the ping command to obtain a 24-hour average. Then we divide the results in half to obtain one-way latency.

• Driving distance: Firstly we determine the driving distance between sites (in kilometers), and multiply the distance by 5 microseconds. Secondly, if the network is point-to-point, we multiply the result by 1.5. If the network is routed, we multiply the result by 2.25 to account for routing delays.

The approximate effect of inter-site latency and available bandwidth can be calculated based on replication throughput for specific link technologies and application write size. The distance and its relationship to cost and performance are major concerns.

2.4 Cost

The cost may prohibit the selection of favorite site distance and feasible solution type. If the required bandwidth proves too costly, we consider moving the remote site closer to the local site or replicating only the most critical data, such as transaction or retransmission logs. Distance and replication throughputs are variables in our analytics. They are determined by customer needs and properties of our application. Once these two parameters have been decided, they are constants. The variables effecting performance (bandwidth and latency) are link technologies and replication solutions.

The cost associated with transmission lines increases with performance requirements. The higher performance is required, the more advanced link technology is used, which increases the cost.

If the lower inter-site latency costs too much to invest in connections between two sites, we consider another replication solution to improve the performance.

3 Characteristics of our application

3.1 Architecture overview

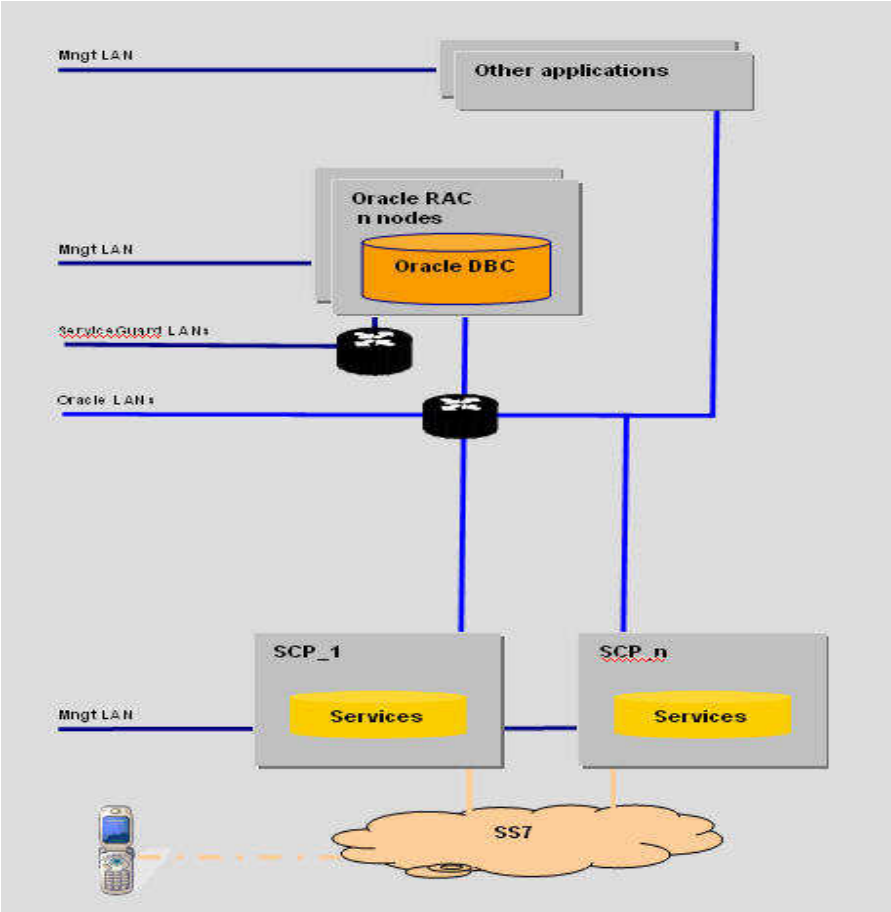

Figure 1. Architecture overview [Ahonen, 2004a].

Figure 1 shows the intelligent network architecture. There are two important network elements in our intelligent network. The first one is a database cluster. It comprises n nodes hosting an Oracle database cluster (DBC), which stores subscribers' data. The second one is a set of servers; each called a Service Control Point (SCP), which manages subscribers' calls. There can be many SCPs in a network [Ahonen, 2004a].

High available features are shown by many behaviors in current architecture. Calls can be handled by all SCPs. Subscribers are not "attached" to a dedicated SCP. Each subscriber is attached to a list of SCPs. If the first SCP in the list is not available, then the call will be routed to the second one. In case of a failure of an Oracle DBC node, the associated connection (SCP/External Applications) will be routed to a survival node [Ahonen, 2004b]. In case of a LAN failure, as all LANs are a pair of cables, the traffic switches to the standby cable.

Figure 2 shows the components of the DB cluster.

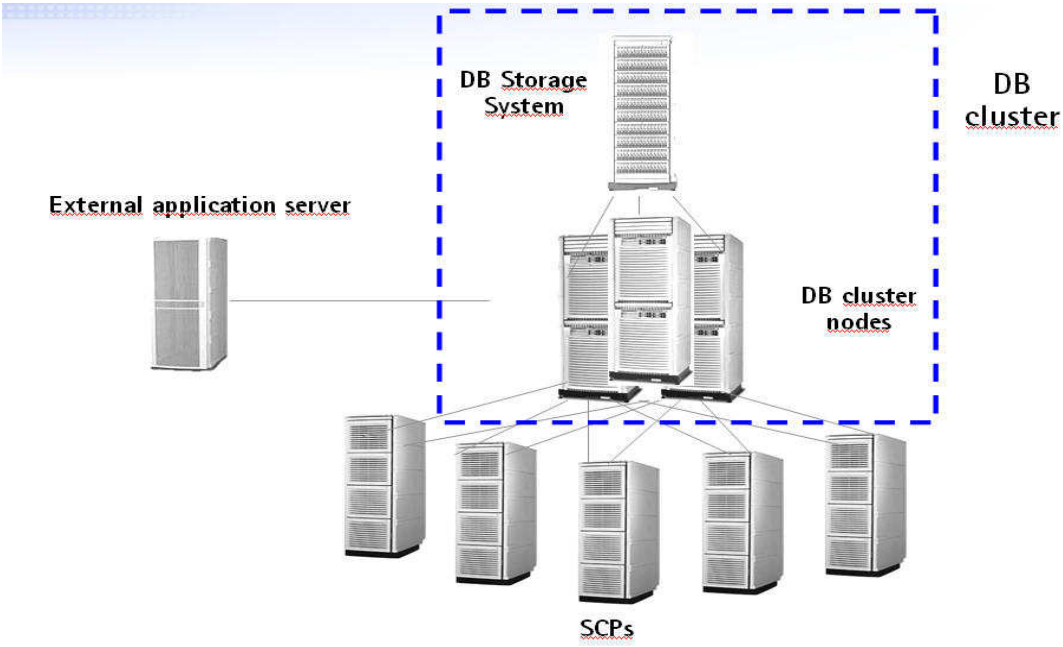

Figure 2. DB cluster overview [Ahonen, 2004a].

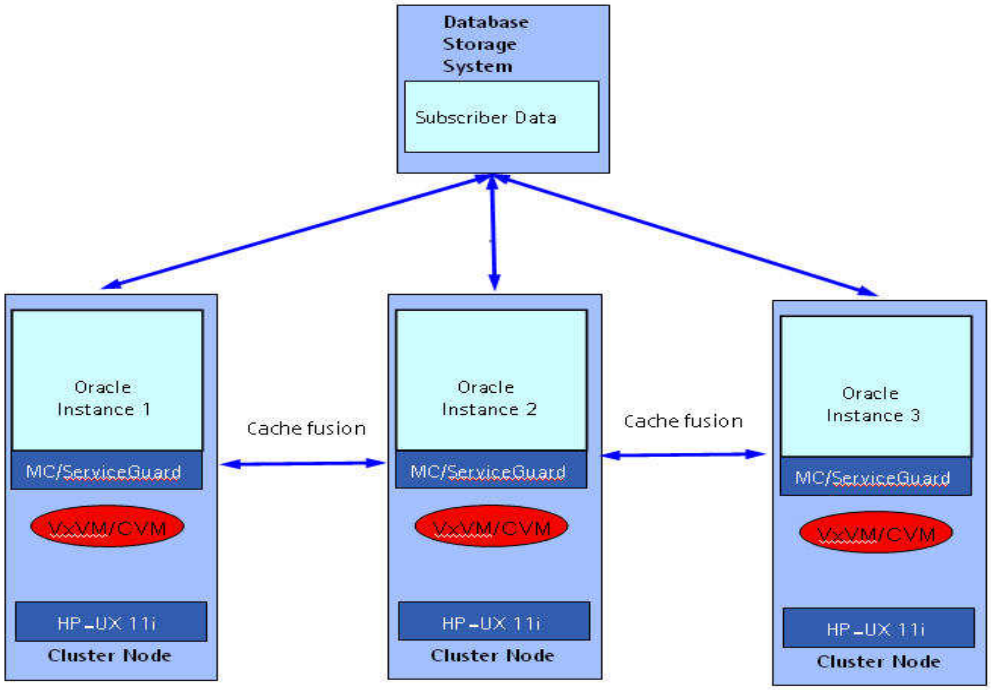

Figure 3. Oracle RAC on DB cluster [Ahonen, 2004a].

Figure 3 lists the software/application running on each node. Oracle 9i Real Application Clusters (RAC) is an option of the Oracle 9i Database that allows running multiple parallel servers in a cluster. The database resides on a highly available disk array that is shared between the servers and instances [Bauer, 2002a]. Each DB cluster nodes runs HP operating system (HP_UX 11i), volume management application (VxVM/CVM), and cluster software (MC Serviceguard). There is only one copy of data residing on database storage which is shared by two or more Oracle instances running on two or more cluster nodes. One of the key innovations in Oracle 9i RAC is that it introduces the full cache fusion. High speed memory to memory data passage between clustered nodes is implemented via low-latency cache-to-cache communications over industry standard interconnect technologies [Bauer, 2002b].

Subscribers' data in Oracle DBC are very important in call processing. There is an application running on SCP, which provides the methods to retrieve all the needed subscriber data into SCP and to update the modified data back into the database. These access methods can be divided into two categories: data retrieval and data update. The methods are implemented by utilizing HP SQL Access interface. The most frequently used database operations are insert, delete and update. [Kemppinen, 2004]

3.2 Current RAC limitation in disaster tolerance

The standard failure resistant Oracle RAC is composed of n nodes, in Active/Active configuration, but located in the same rack. Such a solution is resilient to any kind of failure that does not involve all nodes simultaneously, but is not resilient to an entire site failure [HP, 2004]. Thus high availability is guaranteed only at one site. When we consider the natural disaster happening at one site which can cause a halt of the whole service, a secondary database at another remote site is introduced. This secondary database continuously replicates the data from the primary database, by which the failover can be made easily and immediately in case of breakdown of the primary site.

There is a solution called "RAC on Extended Distance Clusters" available in the market [HP, 2004]. We will describe this solution later in the thesis. However, this solution requires that these two sites are connected by using dense wave division multiplexing (DWDM) equipment and dark fibers [HP, 2004]. DWDM is an optoelectronic technology using dark fiber, which is very expensive. Not many operators can invest in such an expensive solution. The target of this thesis is to find out cost-efficient solutions.

4 Requirements analysis

Literally the general requirement from a customer is to find a solution that would avoid site disaster failures, such as fire, flood, or anything that would destroy an entire operating site. Another site is introduced into the structure to be a backup. We call this a secondary site in order to differentiate it from the primary site. These two sites are backup sites to each other. They provide traffic handling on both sites when there is no disaster, with data replication to each other. Whenever a disaster happens on one site, traffic will be routed to the other site and the whole procedure is transparent to end user. Breaking down the requirements into small pieces helps us to understand them easily:

- The solution should be resilient to a whole site failure. (Disaster destroys the whole datacenter).
- The overall solution is composed on two identical operating sites. Each site is composed with exactly the same HW.
- The solution should provide an Active/Active Architecture. The two sites should be active simultaneously handling traffic when there is no disaster.
- In a case of disaster, the entire site activity (traffic, provisioning, etc.) switchover needs to be as faster as possible, with minimal data loss. The automatic failover with short downtime is the key requirement in our business case.
- In a site disaster tolerance solution the traffic is split between the two sites: each site manages 50% of traffic. This consideration is due to the fact that traffic handling at one site will double when a disaster happens.
- In a case of failure of site1 (or site2): From an SCP point of view calls will be automatically routed to the SCPs in the survived site because subscribers are not "attached" to a specific SCP. From an Oracle DBC point of view, **s**ubscribers' data are replicated/synchronized between two sites. The other site is able to manage calls, which were previously handled by the failed site. The key point is to ensure that the provided service is available and equivalent at all times.
- The solution should be cost-efficient. We are looking for a reasonable and feasible solution, instead of a perfect solution with huge cost.

The location of the remote site and the data transferred between two sites are variables in our requirement. Different customers have different amount of data to be transferred and their preference on distance between two sites. We can not offer a single solution that can fit every customer. Thus, distance and its relationship to cost and performance are our

major concerns. The purpose of this thesis is to present a generic integrated solution for medium or large business who has huge amount of data (replication throughput can be tens or hundreds of Megabytes per second) to be transferred over long distance (hundreds or thousands of kilometers between tow sites) by taking into account the cost and performance.

We are bound to use Oracle database technology (as mentioned in Chapter 3, our server runs Oracle database) and thereby, we take into account the constraints associated with the Oracle DBC. Two sites to tolerate site disaster mean two active Oracle DBCs to be maintained at the same time. The data must be replicated from each other to keep the consistence between the two sites. Any changes in one database will be timely synchronized to the other one in a continuous, nearly instantaneous manner. Figure 4 shows the draft architecture of the solution.

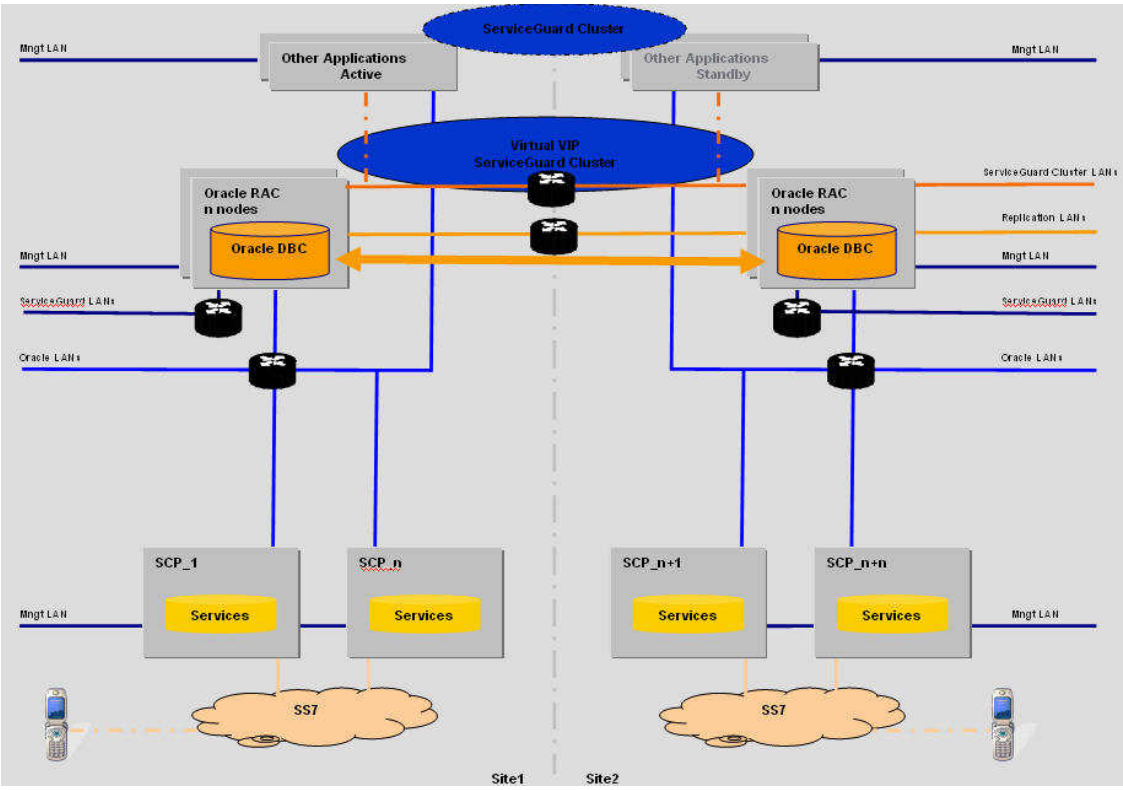

Figure 4. Disaster tolerance solution [Cuttaz, 2006].

5 Solutions evaluation and integration

5.1 Existing replication and synchronization technologies

The existing database can be fragmented and duplicated. The copies are kept in sync by a set of applications. But implementing our own application is not an adaptable option due

to the massive implementation efforts. There are many existing solutions to provide data protection strategies in order to recover the data in a timely manner when needed. We start by evaluating the technologies and further integrating the existing mature solutions into our own solution.

5.1.1 Backup and Recovery

Normally the well-designed and integrated "Backup and Recovery" strategy is needed in database deployment. The backup can be local or remote copies of data recorded on high-speed tape or disks. The backups can be done online or offline, including full or incremental backup data. There are many levels of backup available nowadays. File system backup and database backup are the most useful and popular ones. The backup solution takes too long to recover from failure without any automatic failover, which can not fulfill the key requirement in our business case. [Laurila, 2005]

5.1.2 Snapshots

Snapshots are images of all or part of a disk file system that is taken periodically and stored in another disk allocation [HP, 2005b]. Rather than going back to the previous night's tape backup, we can restore the earliest snapshot in which the corruption does not exist after database corruption happened. This technology is periodically providing a static backup of the file system, instead of an active replica site [Seikku, 2004].

5.1.3 RAID (Redundant Array of Independent Disks)

The fundamental principle of RAID is to use multiple hard disk drives in an array to mirror the data in different places. RAID array provides data security, fault tolerance, improved availability and performance, increased and integrated capacity. RAID array can be configured in many ways including as a single unit or in various combinations of striped and mirrored configurations [HP, 2005a]. This technology is normally used in an array at one site. Therefore, it is not an applicable technology for multiple sites environment.

5.1.4 Remote Data Mirroring

Remote data mirroring is an array-based solution that replicates data by sending track-by-track changes from a primary site to a remote secondary storage subsystem over a secure network. In the event of an outage or disaster on the primary site, the database may be

restored and recovered at the mirrored site. This one can be considered as an applicable technology in our solution.

5.1.5 Data Replication

Data replication is a software-based solution that copies data from the primary database to one or more secondary database. These multiple databases together constitute a distributed database system [Özsu and Valduriez, 1999]. The transactions can be replicated continuously or on a scheduled basis. The strategy of resolving conflicting transactions that appear on the same dataset at the same time but in different database is a topic in our discussion**.** This method can be considered as an applicable technology in our solution.

There are many existing mature solutions in the market, e.g., Oracle RAC, Oracle Stream, Oracle Replication, and Oracle Advanced Replication. Oracle RAC can be configured on extended distant clusters separated from each other longest by 100 km. As distance increases it will slow down both cache fusion and I/O activity in RAC. [Peterson, 2006] The impact of this will vary by applications. Replication via Oracle Streams would require very good knowledge about Oracle Streams and a much more complicated hardware configuration [McElroy and Pratt, 2005]. Oracle Streams also allows the replication of a subset of the tables on the source database to the target database [McElroy et al., 2005]. This ensures that only the data needed to be protected is transmitted across the network, especially when the available network bandwidth is not enough to keep up with the redo-log generation rate [Urbano, 2003b]. The Oracle Advanced Replication [Burroughs, 2002] is an improved replication way over Oracle Replication, which will be discussed in the thesis.

5.1.6 Automated Standby Database

Standby database provides a completely automated framework by maintaining transactionally consistent copies of the primary database. There are two manners to transmit the changes from the primary database to the standby database. The synchronous manner can enable zero data loss but having the impact on performance. On the contrary, the asynchronous manner minimizes the potential performance impact by bringing the risk of losing the newest data just changed on primary site. It is an effective means for disaster recovery by providing an automated framework to switch over to the standby system in the event of corruption on the primary site [To and Meeks, 2006]. This one can be considered as an applicable technology in our solution by some slight modifications.

Oracle Data Guard [Oracle, 2007] proved to be a key technology component providing standby database solution in many organizations' data protection scheme. Additionally considering that Oracle database is already deployed in our intelligent network, we will choose Oracle Data Guard as a basic when proposing our integrated solution in Section 5.4.

5.2 Properties and challenges of replication solutions

5.2.1 Distributed database

Distributed database is an important concept in our discussion. Whatever technology or solution we choose, we have to formulate a distributed database across two sites. A distributed database is defined as a collection of multiple, logically interrelated databases distributed over a computer network. The "logically interrelated" and "distributed over a computer network" are the two important terms in this definition. A distributed database system is a "collection of files" that can be logically related, but also structured among these files. They should be accessed via a common interface. [Özsu and Valduriez, 1999]

It is usually desirable to be able to distribute data in a replication manner across the machines on a network. The performance will be increased dramatically by replication since the diverse and conflicting user requirements can be more easily accommodated. The user can access the data at any working site without the limitation to the local site. This increases the locality of reference. Furthermore, if one of the machines fails, a copy of the data is still available on another machine on the network. [Özsu and Valduriez, 1999]

However, the replication causes problems in updating databases. The decision whether to replicate or not, and how many copies of any database object to have, depends on a considerable degree of user applications. The more predominantly update-oriented the application, the less replication we should have. From the user's perspective, it is wise to act as if there is a single copy of the data and neglect the existence of the other copies. It is desirable that the database management system should provide the replication transparency as a standard feature to user applications. Replication transparency refers only to the existence of replicas, not to their actual location. [Özsu and Valduriez, 1999]

5.2.2 Synchronous and asynchronous replication mode

Replication is the process of copying and maintaining database objects in multiple databases that make up a distributed database system. Changes applied at one site are

captured and stored locally before being forwarded and applied at each of the remote locations. [Urbano, 2003a]

There are two replication modes existing. One is synchronous and the other one is asynchronous. In asynchronous mode, information about data changes on one database is captured and stored in the deferred transaction queues and they are propagated and applied on the second database at regular intervals. The interval can be controlled by the user. The changes that happened on the local table might already be committed and however lost at the remote site when replication fails.

There is also synchronous replication available. Synchronous mode ensures that the change is successfully applied at both the local site and at all replicated sites. Otherwise the transaction is rolled back. Synchronous replication is most useful in situations where users have a stable network and require that their replicated sites remain continuously synchronized. But we can not use it in our solution because our network environment does not make synchronous mode operate smoothly.

An update of a table results in the immediate replication of the update at the other participating master sites in synchronous mode. Therefore, if the master site cannot process a transaction for any reason, then the transaction is rolled back at the master sites. This is a too demanding requirement in our environment. We also see the latency added by synchronous mode in the following discussion. However, it is possible to configure asynchronous replication so that it simulates synchronous replication by using "scheduling continuous pushes" [Pratt, 2001].

We take the array-based replication software as an example to describe the procedures of synchronous and asynchronous modes. The key which can differentiate whether it is synchronous or asynchronous mode is that the changes are replicated to remote sites right away before committed at the local site or in a delayed manner after being committed at the local site.

In HP Continuous Access EVA software, the source array acknowledges I/O completion after replicating the data on the remote array in a synchronous mode. Synchronous replication prioritizes data currency over response time. HP [2006] describes the following procedure to complete the replication in a synchronous mode:

1. A local (source) array controller receives data from a host and stores it in cache.

2. The local array controller replicates the data to the remote (destination) array controller.

3. The remote array controller stores the data in a virtual disk on the Disaster Recovery (DR) group and acknowledges I/O completion to the local controller.

4. The local array controller acknowledges I/O completion to the host.

5. The write is flushed from cache.

Synchronous replication has no need for a write pending queue. Writes remain in Small Computer System Interface (SCSI) command queue in the host port until acknowledged by the local and remote arrays. [HP, 2006]

In an asynchronous mode, the source array acknowledges I/O completion before replicating the data on the remote array. Asynchronous replication prioritizes response time over data currency. The following is a procedure to complete the replication in an asynchronous mode [HP, 2006]:

1. A local (source) array controller receives data from a host and stores it in cache.

2. The local array controller acknowledges I/O completion to the host.

3. The local array controller sends a replication of the data to the remote (destination) array controller.

4. The remote controller stores the data in cache.

5. The remote array controller writes the data to the virtual disks in the destination DR group and acknowledges I/O completion to the local controller.

6. The local controller flushes the data from its cache.

The maximum size of the “write” pending queue limits asynchronous performance. With a small “write” pending queue, lower bandwidths struggle to support applications with erratic or high peak load rates. “Writes” fail when the pending queue is full.

For software-based replication, the control point is not I/O completion acknowledgement. Instead, it is transaction commitment. Otherwise, the procedure is the same.

From the above description we learn that there will not be any replication conflicts in synchronous mode because every write operation can only be completed by applying the same “updates” to the replica site. But there is another serious problem, the I/O latency problem. If any reason preventing the data is replicated to the remote site exists, e.g., the same row of data is locked on the remote site, network connection between two sites is too slow or fails, bandwidth between two sites is low, traffic jam in the connection, etc., then the application which is executing the write at local site has to wait until the

replication is finished. This kind of waiting might cause application timeout and calls are dropped. Some operating systems, such as HP-UX, have a limited I/O queue depth and are expected to spread I/O across many physical disks. Replicating data for a high-performance application on an operating system with limited I/O queue depth can significantly affect performance when synchronous mode is used.

The recovery point objective (RPO) is the amount of data loss that the business can tolerate as a result of a disaster or other unplanned event requiring failover [HP, 2006]. RPO is measured in time and ranges from zero to several hours. An RPO of zero means no completed transaction is lost and requires synchronous replication.

Synchronous mode provides greater data protection (RPO equals zero). Asynchronous mode provides faster response to server I/O. The choice has implications for the required bandwidth of the inter-site link. In general, synchronous mode requires higher bandwidth than asynchronous mode does. In some instances, synchronous mode can require twice the bandwidth for average workloads and ten times the bandwidth for peak loads.

With synchronous replication, the inter-site link must accommodate the peak write rate of our applications. Insufficient replication bandwidth impacts user response time, RPO, or both.

When we determine which technology is most effective, we have to compare the average load and peak writing rate of our applications with the capacity of inter-site link technologies. HP [2006] suggests that the average load on any link must not exceed 40% of rated capacity during normal operations, and the peak loading must not exceed 45% of rated capacity.

Based on the above analysis we know that the synchronous mode of replication is not feasible in heavy-load system. It demands a stable network environment and very expensive high-speed connections between the two sites. The latency must be estimated by calculating the history load data and compared to the application timeout setting. The inter-site link technology is decided by the system load. The cost is determined by the chosen technology.

Figure 5 shows the saturation of synchronous mode and asynchronous mode. We see that asynchronous mode and synchronous mode saturate at approximately the same rate. Asynchronous mode provides the quickest host I/O response time without additional throughput or performance. Nevertheless synchronous mode offers the highest data consistency.

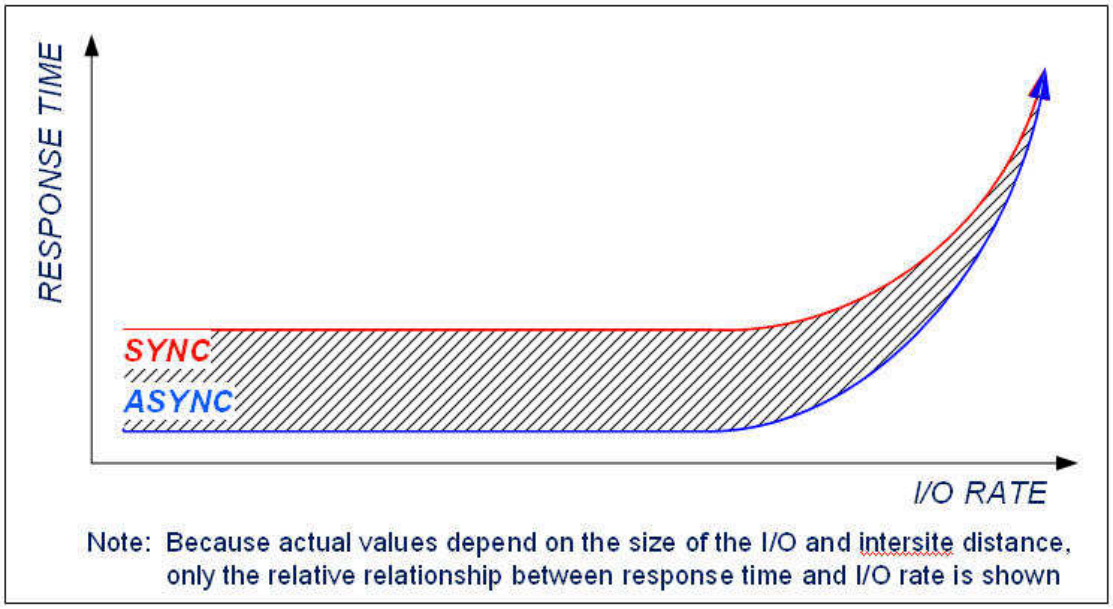

Figure 5. Asynchronous versus synchronous replication saturation [Seikku, 2004].

5.2.3 Array-based and software-based replication

Software-based solution allows the storage to be laid out in a different fashion from the primary site. In theory, customers can put the files on different disks, volumes, file systems, etc. Primary and secondary storage systems do not have to be identically configured. The two storages at two sites can be from two different vendors [Oracle, 2006]. However, most companies choose to configure the two sites identically in order to simplify the management in practice.

On the contrary, array-based solutions are restrictive in the sense that many of them are proprietary and the secondary site can only use the identically configured storage systems from the same vendor that manufactures the primary site [Oracle, 2006]. This restriction is an important point to be bear in mind when we choose the configurations of the disaster recovery sites. Customers quite often choose another vendor as the storage for its disaster recovery site for business reasons.

Some deep knowledge of array technology is needed to plan the disk groups inside an array and the database residing on the array in array-based replication. For instance, there are some special requirements about how to configure the virtual disks in HP Continuous Access EVA. A data replication (DR) group is a logical group of virtual disks in a remote replication relationship with a corresponding group on another array. Hosts write data to

the virtual disks in the source DR group, and the array copies the data to the virtual disks in the destination DR group [HP, 2006]. The DR group in one array can have relationship with multiple arrays.

Since we consider these two sites to be replicable to each other, the bidirectional replication is used. In bidirectional replication, an array can have both source and destination virtual disks, provided that the virtual disks are in separate DR groups. (One virtual disk cannot be both a source and a destination.) For example, one DR group can replicate data from array A to array B, and another DR group can replicate data from array B to array A. Obviously the same table in database can not reside on the source and destination virtual disks at the same time. Otherwise the "updates" of the same table on each site would not be able to be replicated to each other. Disk groups on arrays in a bidirectional relationship should be the same size and type. Bidirectional replication enables us to use both arrays as primary storages while they provide disaster protection for another site. With arrays that support a maximum of eight source-destination pairs per DR group, we also have to reconfigure Oracle database to reduce the number of virtual disks to eight. [HP, 2006]

The distance and configuration size on the management of arrays at remote site will affect the replication performance in array-based replication. We have to estimate configuration size, identify each disk group, disk drive, defined server, virtual disk, DR group, and source-destination pair as an object to be managed. As the number of objects increases, so does the time it takes to discover the objects and manage the array. Similarly, the longer distance between two sites, the more time it takes to complete tasks. A configuration that has many objects and extreme distance between two sites would require unacceptable management time. The management tasks require bandwidth which may be already dedicated to replication processes.

In synchronous mode, applications cannot proceed to the next transaction until the data from the current committed transaction is written to disk at the remote location. Application performance is thus affected by the time it takes to transmit data from the primary site to the remote location (i.e. network latency), write it to disk (disk I/O), and receive return acknowledgement from the remote site (network latency) that the data has been received [Oracle, 2006].

Some of the software-based solutions only transmit writes to the redo logs of the primary database, whereas array-based solutions must transmit these writes as well as every write to data files, additional members of online log file groups, archived log files, and control

files [Oracle, 2006]. Array-based solutions do impact replication performance because they subject more data to transmit, inducing delays inherent to synchronous configurations.

Overall, software-based solutions are simpler compared to array-based solutions. Normally they have an easy-to-use graphical user interface for managing data replication without deep knowledge of configuring an array. Besides they have less data to transmit over network.

5.2.4 Technologies of inter-site connection

The supported transmission distance varies with the technology. The price varies according to the chosen technology. Some of them are very expensive and not acceptable by some customers. Cost is an important factor when customer chooses transmission technology. The followings are mature and commonly used technologies belonging to different price level:

- Basic fiber supports a maximum of 500 meters at 1 Giga bytes per second. Shorter lengths are supported at higher bandwidths. The distance varies with the speed of the link [HP, 2006].
- Fiber with long-distance and very long-distance gigabit interface converters (GBICs) and Small Form-factor Pluggable (SFPs) supports up to 200 times the basic fiber distance [HP, 2006].
- Wavelength Division Multiplexing (WDM) is a technology that uses multiple lasers and transmits several wavelengths of light simultaneously over a single optical fiber. Each signal travels within its unique color band, which is modulated by the data (text, voice, video, etc.). WDM enables the existing fiber infrastructure of the telephone companies and other carriers to be dramatically increased. Fiber with WDM supports up to 500 kilometers. The difference between WDM and basic fiber configurations is the addition of a multiplex unit on both sides of the inter-site link. A WDM installation must conform to vendor specifications. Some switch vendors may limit the maximum distance between sites. Performance is affected by extreme distance and limited buffer-to-buffer credits on the fiber channel switch. Connecting the switch to the WDM unit typically requires one switch-to-WDM interface cable per wavelength of multimode fiber. Switches may require an extended fabric license. [HP, 2006]

- Fiber Channel-to-IP and Fiber Channel-to-SONET gateways support the longest distances. The remote replication configuration over IP is similar to the basic remote replication configuration over fiber with the addition of Fiber Channel-to-IP gateways. Fiber Channel-to-SONET configuration is similar to Fiber Channel-to-IP configuration except that Fiber Channel-to-IP gateways are replaced with Fiber Channel-to-SONET gateways. [HP, 2006]
- Dense Wave Division Multiplexing (DWDM) is an optoelectronic technology which can simultaneously transmit multiple separate optical signals through a single optical fiber which is thinner than a human hair. The maximum distance allowed between a DWDM devices pair depends on the particular DWDM vendor product used. But it can reach distances as high as the 100 to 120 kilometers range, supporting more than 150 wavelengths, carrying up to 10 gigabytes per second. Such system provides more than a terabit per second of data transmission on one optical strand. Nevertheless, DWDM is too demanding for most operators because of the expensive investment. [HP, 2004]

Network bandwidth management is not a one-off exercise. It needs careful planning, reviewing and understanding of Service Level Agreements (SLAs) for the supported applications, as well as continuous monitoring of the network to ensure that the business operation goals and availability requirements are being met. The aim is to achieve comparable reliability on commodity-priced hardware and network connections.

5.2.5 Replication conflict problems

Conflicts might happen when there is replication between two sites. When it happens, we focus on finding out a sophisticated conflict detection mechanism and a comprehensive set of automated conflict resolution routines to ensure data convergence throughout the replicated environment [Pratt, 2001].

Conflict detection enables a replication solution to detect changes made to a row in one replica before a previous change made to that row in another replica has time to propagate to the database where the subsequent change was made. In a replication scenario where replicas may be disconnected or only periodically synchronized, the possibility of a conflict increases.

Conflict resolution routines are automatically invoked when a conflict is detected. These are typically routines like one site wins, latest timestamp wins (requires a timestamp

column), or make the changes additive. Custom conflict resolution methods allow customers to expand this capability by writing their own conflict resolution routine.

Replication conflicts occur in a replication environment that permits concurrent updates to the same data at multiple sites. For example, when two transactions originating from different sites update the same row at nearly the same time, a conflict can occur [Urbano, 2003a]. In general, the first choice should be to design a replication environment that avoids the possibility of conflicts. However, there are some parts of data that might be updatable at multiple sites at any time. We have the subscriber "Account" table in two separate sites. For instance, when father and daughter are using the same account to make calls at different sites nearly at the same time, the conflict happens. The application handling the calls or doing the top-up at different sites will try to allocate the money or increase the money from "Account" table using the stored procedure. The possibility of such replication conflict is not so high but still effecting the reputation of an operator if the system makes the money disappear for unknown reasons.

5.2.5.1 Three types of replication conflicts

There are three types of replication conflicts. They are update conflict, uniqueness conflict and delete conflict.

An update conflict occurs when the replication of an update to a row conflicts with another update to the same row. An update conflict can happen when two transactions originating from different sites update the same row at nearly the same time. [Urbano, 2003a]

A uniqueness conflict occurs when the replication of a row attempts to violate entity integrity, such as a `PRIMARY KEY` or `UNIQUE` constraint. For example, when two transactions originating from two different sites insert a row into a respective table replica with the same primary key value, replication of the transactions causes a uniqueness conflict. [Urbano, 2003a]

A delete conflict occurs when two transactions originate from different sites, with one transaction deleting a row and another transaction updating or deleting the same row, which does not exist to be either updated or deleted after the replication. [Urbano, 2003a]

In our application, most of the replicated data are required to be updatable at all replication sites and only small fractions of data are read-only. Under such circumstance we must determine how to detect and resolve replication conflicts when they occur so that the integrity of replicated data remains intact. Nevertheless there are some

environments where conflict detection and resolution are feasible in some cases but not possible in others.

5.2.5.2 Conflicts detection

Corresponding to the three different types of conflicts, there are three different ways of detections. The receiving site detects an update conflict if there is any difference between the old values of the replicated row (the values before the modification) and the current values of the same row at the receiving site. The receiving site detects a uniqueness conflict if a uniqueness constraint violation occurs during an `INSERT` or `UPDATE` of a replicated row. A delete conflict is detected if the receiving site cannot find a row for an `UPDATE` or `DELETE` statement because the primary key of the row does not exist. [Urbano, 2003a]

5.2.5.3 Conflicts avoidance

Even though there are some powerful methods for resolving data conflicts, our first choice is still to design a replication environment that avoids the possibilities of conflicts. By using several techniques, we can avoid conflicts in a large percentage of the data that is replicated. Defining column groups in Oracle database can be an example method to avoid conflicts even if there is no conflict resolution methods applied to the column groups. When a table containing multiple column groups is replicated, each group is viewed independently when analyzing updates for conflicts. [Urbano, 2003a]

For example, consider a replicated table with column group `A` and column group `B`. Column group `A` contains the columns `a1`, `a2`, and `a3`, and column group `B` contains the columns `b1`, `b2`, and `b3`.

The following updates occur at replication sites `S_A` and `S_B`:

- User `UA` updates column `a1` in a row at `S_A`.
- At exactly the same time, user `UB` updates column `b2` in the same row at `S_B`.

In this case, no conflicts result because Oracle analyzes the updates separately in column groups `A` and `B`. If, however, column groups `A` and `B` did not exist, then all of the columns in the table would be in the same column group, and a conflict would have resulted. Also, with the column groups in place, if user `UB` had updated column `a3` instead of column `b2`, then a conflict would have resulted, because both `a1` and `a3` are in the `A` column group.

There are some simple techniques to avoid the above mentioned three types of conflicts. In order to avoid “uniqueness conflicts”, we can append a unique site identifier as part of

a composite primary key. In Oracle database, we have the possibility to select a globally unique value by using the `SYS_GUID` function. Using the selected value as the primary key (or unique) value will globally avoid uniqueness conflicts. For the "delete conflicts", there is one general rule that applications which operate within an asynchronous, shared ownership data model should not delete rows using `DELETE` statements. Instead, applications should mark rows for deletion and then configure the system to periodically purge logically deleted rows using procedural replication. After elimination of the possibility of uniqueness and delete conflicts in a replication system, the number of "update conflicts" that are possible should be limited as well. However, "update conflicts" cannot be avoided in all cases. In case that not all "update conflicts" can be avoided, then we can still try to understand exactly what types of replication conflicts are possible and then configure the system to resolve conflicts when they occur. [Urbano, 2003a]

5.2.5.4 Conflict resolution

After a conflict has been detected, we resolve the conflict with the goal of data convergence across two sites. Normally, most conflict resolution methods work with any data type. We will choose the Oracle as an example to describe different aspects of some pre-built conflict resolution methods.

To automate the "conflict resolution", Oracle provides several pre-built conflict resolution methods to resolve update conflicts. These methods can guarantee data convergence across a variety of replication environments in many situations. Oracle also offers several conflict resolution methods to handle uniqueness conflicts, though these methods cannot guarantee data convergence. Oracle does not provide any pre-built conflict resolution methods to handle delete conflicts. Oracle does, however, allow us to build our own conflict resolution method to resolve data conflicts specific to our business rules. If we do build a conflict resolution method that cannot guarantee data convergence, which is likely for uniqueness or delete conflict, then we should also build a notification facility to notify the database administrator so that data convergence can be manually achieved.

Whether a pre-built or user-defined conflict resolution method is used, it is applied as soon as the conflict is detected. If we have not defined any conflict resolution methods or the defined conflict resolution method cannot resolve the conflict, then the conflict is logged in the error queue.

The Table 1 lists most of the common "update conflict" resolution methods.

Resolution Methods	Explanations
Latest timestamp	Take the value with latest (newest) timestamp
Overwrite	Replace the current value with the new value
Additive	Current value = current value + (new value – old value)
Average	Current value = (current value + new value) / 2
Discard	Ignores the values from the originating site
Earliest timestamp	Take the value with earliest (oldest) timestamp
Maximum	Compare the new value from originating site with the current value from the destination site for a designated column. Take the maximum from them. (column values must always increase)
Minimum	Compare the new value from originating site with the current value from the destination site for a designated column. Take the minimum from them. (column values must always decrease)
Priority group	Take the value from the table with the higher priority.(assign a priority level to each possible value of a particular column.)

Table 1. Update conflicts resolution methods [Pratt, 2001].

Although we have so many pre-built "update conflict" resolution methods, the "latest timestamp" and the "overwrite" conflict resolution methods are the most commonly implemented resolution methods.

The "latest timestamp" method resolves a conflict based on the most recent update, as identified by the timestamp of when the update occurred.

To use the timestamp method, a column in the replicated table of type `DATE` must be designated. When an application updates any columns in a column group, the application must also update the value of the designated timestamp column with the local system date. Because time is always increasing, it is one of the few conflict resolution methods that can guarantee data convergence with most of the update conflicts. For example, the application can maintain subscriber address information at different databases in a replication environment. The information with the newest timestamp is always the updated information which should be correct.

The other method we commonly use to solve "update conflict" is the "overwrite" method. The "overwrite" method replaces the current value at the destination site with the new value from the originating site. It can only guarantee the data convergence for a

replication environment that has a single master site with many slave sites. It is ideal for mass deployment environments which keep all the slave sites in sync with one master site.

Oracle provides three pre-built methods for resolving “uniqueness conflicts”. They cannot actually keep the data convergence in a replication environment. Instead, they simply provide techniques for resolving constraint violations. These methods include appending the global site name of the originating site to the column value from the originating site, appending a generated sequence number to the column value from the originating site, and discarding the row value from the originating site.

Oracle does not provide any pre-built methods for resolving “delete conflicts”. We should design our database and front-end application to avoid “delete conflicts”. This goal can be achieved by marking rows for deletion and using procedural replication to purge such marked rows at regular intervals.

To avoid a single point of failure for conflict resolution, an additional conflict resolution method is defined to backup the primary method. For example, in the unlikely event that the “latest timestamp” conflict resolution method cannot resolve a conflict because the timestamps are identical, a “site priority” conflict resolution method can be defined to break the timestamp tie and resolve the data conflict.

However, conflict resolution is often not possible in reservation systems where multiple bookings for the same item are not allowed. Unfortunately our IN system is such a system that reserving the money in the “Account” table is necessary before calls can be connected. Different applications accessing different replicas of the database cannot reserve or update the same row of “Account” table for multiple calls or top-ups happening at the same time because there is no way to resolve such a conflict. Here is the analysis of reservation process and we see clearly why it is impossible to avoid such conflicts.

“Account” information is one of the dynamic subscriber-specific data that must be updated into the external database residing on DB cluster [Soppi, 2004].

After a successful access initiation to a subscriber-specific data all the dynamic subscriber-specific data is not readily accessible directly from the SCP database. Account information is never stored in the SCP database and accessed directly from the external database when needed. [Ahonen, 2004b]

Before SCP instructs the switch to connect a call from “Subscriber A” to “Subscriber B”, it must reserve the money from “Account” table based on the rough estimation of the cost. The procedure is called “Allocate” [Ahonen, 2004b]. After the call is disconnected, the reservation will be released and the “Account” table will be updated according to the actual cost.

The two flows (taken from [Ahonen, 2003]) in Appendix 1 show how the whole “Allocate” function works. When the updating of the “Account” table starts, there will be a transaction which locks the whole row until the transaction is committed or rolled back. This avoids the updating conflict at one site when there are many Oracle instances trying to update the same row of data. It is implemented by Oracle RAC internal mechanism at one site.

“Allocate” function reserves money from the “Account” table. Whenever there are many requests for updating the same row in the database almost at the same time, only one request succeeds because the accounting row was locked by the first taken running session [Soppi, 2004].

By analyzing the stored procedure of “Allocate” function [Nokia SQL script, 2004] in Appendix 2, we simulate the situation when “update conflict” happens. Assume that the two applications on each site allocate the money at the same time. (Allocation is 90 at `Site A`, 70 at `Site B`.) The balance is 100 before updating. After the concurrent updating, there are two inconsistent statuses at these two sites.

At `Site A`:

l_current_credit = 100 -90 = 10

l_reservation1 = 90

l_balance = 10

At `Site B`:

l_current_credit = 100 -70 = 30

l_reservation1 = 70

l_balance = 30

Before the next update happens, if the `Site A` replicates the changes to `Site B`, the unsolvable conflict happens. We cannot use any methods to solve such a conflict or take any one of them as the correct value.

For this kind of unsolvable conflicts, we can only log them into an error queue. How to design a replicated environment to prevent such conflicts in advance becomes more important.

5.3 Examples of adopting existing solutions in the market

5.3.1 HP Continuous Access EVA

HP Continuous Access EVA is an array-based replication component of HP EVA controller software. When this component is licensed, the controller copies data online to a remote array over a Storage Area Network (SAN) in real time. HP Continuous Access EVA is enhanced to perform remote replication and ensures data integrity across sites. HP Continuous Access EVA enables us to build two copies of application data at two sites that are far enough apart to provide disaster tolerance. [Seikku, 2004]

It has two choices of "write" modes as many replication softwares have, which are synchronous mode and asynchronous mode. The choice depends on the relative business needs for data protection. As already analyzed in Chapter 4, we do not have the stable network environment to adopt the synchronous mode. So we adopt the bidirectional replication under asynchronous mode. The configuration of DR groups has been introduced in Chapter 4.

HP Continuous Access EVA supports direct Fiber Channel and extended Fiber Channel-to-IP links ranging in bandwidths from 2.048 Mega bytes per second to more than 4 Giga bytes per second. The longer the distance, the greater the impact inter-site latency has on replication performance. Table 2 shows the inter-site latency inherent to the distance (assume taking the cable at transmission rate of 5 microseconds per kilometer and the synchronous mode to replicate one block.).

One-way delay (ms)	Point-to-point cable distance in km (miles)
1	200 (125)
3	600 (375)
9	1,800 (1,125)
18	3,600 (2,250)
36	7,200 (4,500)
60	12,000 (7,500)
100	20,000 (12,500) current maximum limit

Table 2. Samples of one-way delay [HP, 2006].

HP Continuous Access EVA has an interactive spreadsheet called “Performance Estimator” that calculates the approximate effect of inter-site latency and available bandwidth on replication throughput for specific link technologies and application write size [HP, 2006]. By supplying the latency and application “write” size, the estimator determines the replication I/Os per second (IOPS) and throughput. The result from estimator is very useful for tuning the system in order to achieve better performance.

Overall, HP Continuous Access EVA is feasible to be taken as our solution. The replication conflict problem can be solved by distributing the application data across two sites. Nevertheless, how to spread the database tables over virtual disks becomes critical in adapting this solution. Comprehensive knowledge of an array is needed when planning disk groups, DR groups and DR group logs in advance. Changing the “write” mode later has some limitations and will cause some “cleaning work” on DR group logs. Saving the development effort drives us to look for an easier solution without spending too much time on array planning.

5.3.2 RAC on HP Extended Cluster

Oracle database software has traditionally been configured in such a way that a single copy of the Oracle software running on a single server managed a single database. In this environment, the quality of the database services depends primarily on the quality of the server.

However, in June 2001 Oracle released Oracle9i with Real Application Clusters (RAC). Oracle9i RAC removed this key architectural limitation, making it possible for a collection of database servers to cooperate in the management of a single Oracle database. Thus Oracle Real Application Clusters delivers a higher quality of service at lower cost by clustering database servers. [Slee, 2004]

Oracle 9i RAC allows multiple instances to access a single logical database across multiple servers, with all nodes able to concurrently execute transactions against the same data repository [Mark, 2002a]. Normally there is only one copy of data which is shared by two or more Oracle instances on two or more servers in the same data center. Data can be managed as raw or as clustered file system data. Since accesses to memory take nanoseconds while accesses to disk take milliseconds, the performance of any operation is directly proportional to the location of the data (whether data is in shared memory or on disk). In order to utilize shared memory area, Oracle 9i RAC introduces full Cache Fusion by implementing high speed memory to memory data passage [Cutler et al.,

2005]. Because of Cache Fusion, users can coordinate access so that all servers can modify any of the data. This allows work requests to run on any server, instead of being limited to a specific server because of some "partitioning" algorithm requirements in the earlier days. If a server fails, the surviving servers in the cluster automatically take over processing chores. Oracle RAC can automatically transfer and rebalances workloads from a failed server to surviving servers in a cluster. Oracle RAC provides scalability by introducing additional servers into the cluster in a nondisruptive fashion to help with increasing workloads. Figure 6 shows RAC structure in one data center.

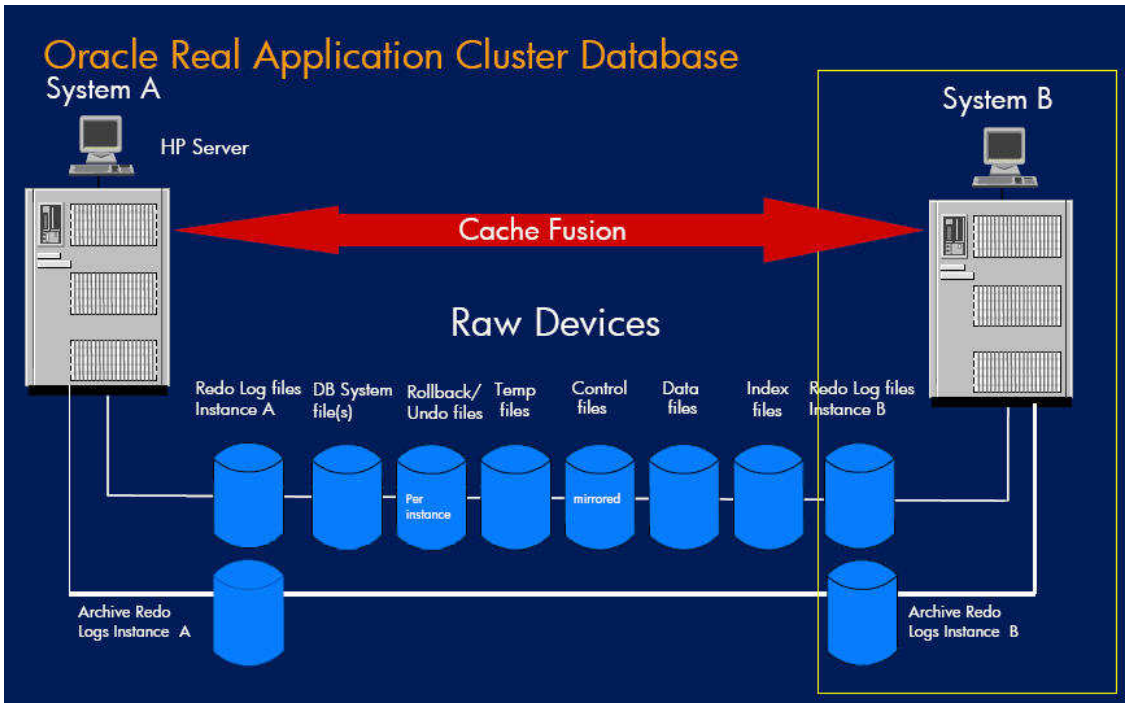

Figure 6. RAC on HP-UX clustered database in one data center [Cutler et al., 2005].

However, RAC in one data center cannot cope with the entire site failure. We are looking for a solution which can extend flexibilities and scalabilities of RAC across two sites to avoid a site disaster. RAC on HP extended cluster give us this possibility to inherit the advantage of RAC across two sites.

HP provides an extended cluster for RAC in order to introduce another data center. When we configure the RAC on HP Extended Cluster, a single logical database instance is split across two data centers, separated by an unprecedented 100 kilometers. Data is replicated and synchronized between two sites, which are functioning as a virtual single entity. Even though there may be up to 100 kilometers between the two sites, the administration

of the application data is equivalent to managing a traditional RAC application located in a single data center. Figure 7 presents the RAC structure on HP Extended Cluster.

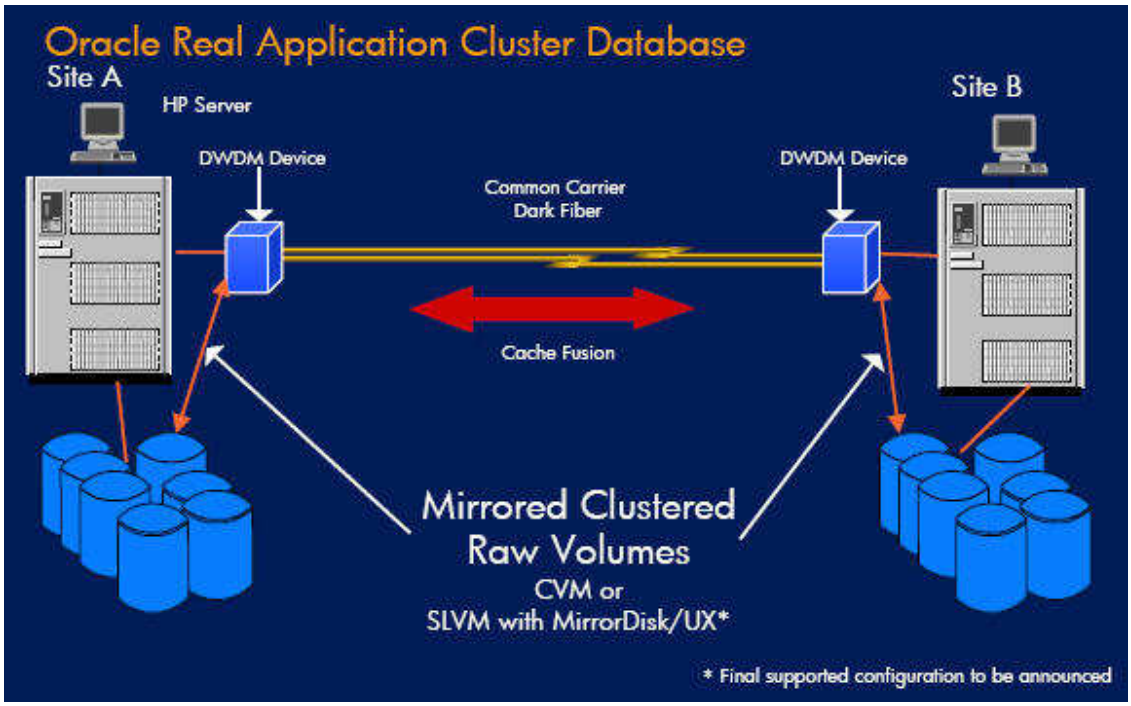

Figure 7. RAC on HP-UX Extended ServiceGuard Clustered Database [Cutler et al., 2005].

Previously RAC shared a single set of storages located on servers in the same data center. Now the two sites are connected by using DWDM equipment and dark fiber. Each data center has its own set of storages which are synchronously mirrored using either CVM mirroring or SLVM with MirrorDiskUX. All traffics including “mirrored writes” and heartbeats between the two sites are carried on the dark fiber. The DWDM and fiber optic media together provide a very low latency interconnection that is essential to RAC performance and scalability.

Database replication is achieved via array-based software mirroring, where RAC synchronizing the database caches via cache fusion. We have two replication mechanisms to choose. The one is using SLVM for volume management and HP XP CA for data replication. The other one is using VxVM/CVM for volume management and EMC SRDF for data replication.

If an entire data center is down, the remaining data center continues to function through re-routing of SCPs to the functioning environment. This capability provides continuous

availability across two sites. Once the failed data center comes back online, all resynchronization takes place automatically and is transparent to end users.

The application is resident on a single data repository so that the administration of the overall environment is simplified. The concept of unnecessary replicated databases and associated burden of replicated management chores, have been eliminated. Even spread across two sites, the Oracle 9i database is still a single database instance, possessing inherent availability, scalability and flexibility [Slee, 2004].

The virtual server environment is permitting both data centers to ostensibly see the data simultaneously and to continually harness the aggregated resources from both sites.

HP has been working on developing the test plan for a RAC on extended cluster. Together with partners including Oracle, AT&T and Nortel, HP's testing focuses on demonstrating the ability to achieve a robust solution capable of maximizing resource utilization across extended distances [Peterson, 2006].

Table 3 shows the various solutions which has been tested and certified with HP ServiceGuard and Oracle.

HP-UX 11.11 (PA-RISC) Oracle 9i RAC and Disaster Tolerance

	Disaster Tolerant RAC Solutions				Extended RAC	
Topology	SGeRAC		Veritas DBeAC	SGeRAC with Continentalclusters	SGeRAC Stretched 2 arrays	Extended SGeRAC 2 arrays*
Distance	Local DC		Local DC	Unlimited	10 kms	100kms
Volume Manager	SLVM	Veritas CVM	Veritas CVM	SLVM	CVM	SLVM CVM
CFS	-	-	VCFS	-	-	-
SW Mirroring	MirrorDisk UX	CVM with VxVM Mirroring	TBD	N/A	CVM with VxVM Mirroring	MirrorDisk UX CVM Mirroring
HW Mirroring	N/A	N/A	N/A	CA XP	N/A	N/A
Storage	VA, XP, EMC, EVA	VA, XP, EMC	XP	XP	VA, XP, EMC	TBD
# Nodes	16	4	TBD	16 per cluster / total of 32 nodes	2 or 4	2
Bi-directional Failover	N/A	N/A	N/A	Yes	Yes	Yes

Table 3. Tested disaster tolerant cluster solutions for Oracle on HP-UX [Cutler et al., 2005].

A series of loading and failure scenarios were executed to validate the expected disaster tolerant characteristics. Each test was structured to induce significant transaction based

traffic, perform the sequenced set of failure inducing events and the subsequent validation of data integrity during the failures.

We took some test cases and results from HP [Cutler et al., 2005]. The hardwares in the tested configurations are as following:

- 2 rp7410s (8 CPUs, 32GB RAM).
- 2 Tachyon XL2 Fiber Channel HBAs.
- 3 Fiber-based GB Ethernet interfaces.
- 4 Ultra SCSI internal disks (73GB).

The following categories of testing were performed:

- Inter-Process Communication (IPC) tests.
- I/O tests.
- Online transaction processing (OLTP) tests.
- Failover tests, including host failure, storage device failure, inter-site failure, and DWDM link failure.

By analyzing the test result provided by HP [Cutler et al., 2005], we get some conclusions about RAC performance on HP Extended Cluster:

- The extended RAC solution over DWDM tests show HA characteristics of traditional RAC by allowing the RAC continue to be available and accessible when failure occurs on one component or one site.
- The overall performance degrades over distances as expected. The shorter distance, the better the performance under the same system load. Once both IPC and I/O tests yield reasonable performance change, we can expect the application would perform in an acceptable manner at the same distance.
- Introducing multiple cluster interconnections is beneficial to improving the overall performance.
- Since performance results will vary depending on our particular configuration and application characteristics, the assessment of the application workload in a local configuration is critical prior to implementation over extended distances. Tuning and consideration of hardware components would improve the application performance at longer distance. Additionally, a variety of performance

enhancements at various levels are available by HP [2004], such as network buffering, workload distribution and database partitioning.

- The trade-off between performance and high availability with disaster tolerance is a key element in a design decision.

Overall, RAC on HP Extended cluster eliminates the current physical restriction of RAC by implementing a single RAC database spanning two geographically distant sites. It combines HA with disaster tolerance and fully utilizes all resources. This solution is simpler than maintaining a standby database because the workloads among cluster nodes across sites are automatically balanced by RAC. Even though the array-based replication is used, there is a single logical database from management point of view. Spreading the database tables over virtual disks are taken care by RAC without too much effort from DBA. The downsides which might prevent us to adopt this solution are the 100 kilometres limitation of the distance between two sites and the high cost of getting DWDM equipment and support.

5.3.3 Oracle Replication

The Oracle Replication is a fully integrated software-based replication feature of the Oracle server. It is automatically installed and upgraded in every Oracle 9i Database. The replication objects which can be replicated in Oracle Replication include tables, indexes, views, triggers, procedures, etc. [Pratt, 2001]. These objects are logically grouped into a collection which is called replication group. By doing this, it is easier to administer many objects together inside one group.

Multi-masters replication is also called peer-to-peer replication, which enables multiple sites (there are two sites in our discussion) acting as equal peers to manage groups of replicated database objects [Burroughs, 2002]. Each site in a multi-masters replication environment is a master site and each site communicates with the other master sites. Data can be accessed and modified from both Oracle databases at each site. We configured the Oracle Replication to work under asynchronous mode as a testing case to see how it works when different replication failures happen.

Using asynchronous replication means that data conflicts are possible because the same row value might be updated at two different master sites at nearly the same time. Oracle provides pre-built mechanisms that can resolve the conflicts. If the pre-built conflict resolution methods cannot resolve, then we have the option of building and using our own conflict resolution methods in accordance with our business rules.

Firstly we simply describe the overview of our configuration. The data (all or a subset) are replicated between the Site 1 Oracle database and the Site 2 Oracle database. The Oracle Replication process, even if it is almost instantaneous, is asynchronous. This means that an operation on one Oracle database is committed before it is actually applied on the second Oracle database. If the two sites update the same row during the time when the replication needs to apply the change at the remote site, Oracle Replication considers that a replication conflict happens. When Oracle Replication detects a conflict, the originating transaction is logged in the replication errors queue and the database administrator is responsible for resolving the data conflict manually if no pre-built conflict resolving method is defined. As the data is replicated between two Oracle databases at two sites, the provisioning of the data by external applications must be done at a single site to only one Oracle database. This provisioning could be done to the Site 1 Oracle database or to the Site 2 Oracle database.

Secondly we monitor how the system behaves in different failures. Scenarios of Oracle database failure, Oracle Replication LANs failure and Oracle Replication processes failure will be discussed.

In case of Oracle completed database failure at one site, the SCPs which connect to the failed database instance will get an error and these SCPs will not be able to handle subscriber calls. In that case, calls will be simply routed to another SCP located on the remote site. Notice that in this failure, the data not yet propagated from the failed site to the survival site (data stored in the replication queues, but not yet applied) will be lost. The Oracle Replication framework on the survival site will be marked as disabled since only one site is left. Oracle transactions from SCPs will not be stored in the replication queues until Oracle Replication framework is recovered. When the failed site resumes, the database administrator must operate to resynchronize the two Oracle databases and restart the Oracle Replication. This kind of failure has no impact on external applications. The external applications connect to the failed Oracle database will switch to the survival Oracle database, thanks to the HP ServiceGuard Virtual IP. **[**Cuttaz, 2006**]**

In case of Oracle Replication LAN failure, there will be two possibilities. If only a single Oracle Replication LAN fails, the replication processes will switch automatically onto the second LAN. There are no serious impacts except a short period that Oracle Replication stop replicating data because Oracle needs 10 minutes to detect the LAN failure. Data are stored into the Oracle Replication queues during this period. If the LAN failure occurs between the “Send” event and the “Acknowledge” event, the replication processes will resent the transactions automatically. In case of double Oracle Replication LANs failure,

the replication processes will not be able to sent transactions to remote database. Transactions will be stopped on both sites if the outage is greater than the *Max_Outage_Time* **[**Cuttaz, 2006**].** All the transactions on Oracle databases at both sites will be stored into the Oracle Replication queues.

When the Oracle Replication LANs connection becomes up, the database administrator needs to restart the Oracle Replication processes at both sites. Accordingly, Oracle Replication at each site sends queued transactions to the other site.

The replication queue can store a maximum of *Max_Trans_Queue* **[**Cuttaz, 2006**]** before becoming full. If a replication queue becomes full, transactions will be rejected. Consequently SCPs and external applications will get errors.

The replication processes can fail in case of unreachable remote Oracle database. There are many reasons, for example, replication LAN failures, Oracle server or Oracle listener failures, Oracle server errors, etc. Even if the Site 1 replication processes are down, transactions on Oracle database at the Site 1 are stored in the Site 1 Oracle Replication queue. These transactions will be sent to the Oracle database at Site 2 when the database administrator has restarted the Oracle replication processes at Site 1. The Oracle replication processes could be down at Site 1, however up on the database at the Site 2. Oracle replication process at Site 2 still can send and apply transactions to the Oracle database at Site 1.

There are some requirements and limitations to be considered when taking Oracle Replication into use. We firstly consider the Oracle connection bandwidth between two sites. The distance between two sites cannot be too close. Otherwise a disaster happen at one site might have impacts to the other one. This contradicts to our intention of designing a disaster tolerance solution. Normally the distance will be 40 kilometers or longer. The Oracle Replication has to reserve a bandwidth of minimum 2 gigabytes per second on each of the two HA connections (on each LAN) between two sites. Similarly the HP ServiceGuard requires reserving a bandwidth of minimum 100 megabytes per second on each LAN. The two HA connections should not go though the same network devices, otherwise this device would represent a single point of failure in the solution.

The two sites can be connected using DWDM equipment and dark fiber. Of course we can use other transmission technologies in between and keep in mind that 2 gigabytes per second is very influential to get acceptable replication performance. This is very important to keep the high performance of Oracle Replication, so that we can reach the target that these two sites are synchronized in a continuous, nearly instantaneous manner.

Secondly we think about Oracle Replication sizing. The sizing of the Oracle Replication environment (Oracle Replication queues, *Max_Trans_Queue, Max_Outage_Time,* etc.) depends on the maximum latency target and the number of operations to be replicated. They are configurable in Oracle Replication.

Thirdly we think about the load of the database server. If all Oracle database nodes are fully loaded, an additional node dedicated for the Oracle Replication could be requested. We must plan the resources (CPU, RAM, I/O) associated to the Oracle Replication in advance when we are integrating this solution.

Finally we look at the performance data of Oracle Replication from Oracle. From their test results and experience, Oracle Replication throughput can be around 1000 operations per second in one direction (2000 in total) when running on HP rp4440 server which has eight 1GHz processor and 16Gb RAM. The resources usage is estimated to be 20% for CPU. **[**Cuttaz, 2006**]**

Overall, Oracle Replication is easier to use than array-based replication solution. As a software-based solution, it requires less array configuration skills to operate. It has the similar manners (synchronous and asynchronous) of replication but less data to be transmitted compared to array-based replication. It can replicate most of our application data without conflicts. However, there are still some "update" conflicts which cannot be solved automatically without administrator's intervention. Furthermore, the high-speed connection between sites required by Oracle Replication is too demanding for some customers. Last but not least, human intervention in re-synchronizing the two Oracle database after disaster is a weakness compared to RAC on HP extended cluster solution.

5.4 Integrated solution based on Oracle Data Guard

5.4.1 Data Guard Introduction

Oracle Data Guard is a standard feature of Oracle Database Enterprise Edition and is integrated with other Oracle features to provide a comprehensive HA/DR solution to protect mission critical data. Oracle Data Guard ensures high availability, data protection and disaster recovery with the means of one active database and one or several standby databases. It runs on any hardware platform which Oracle supports. It doesn't require proprietary storage subsystems, special and costly network devices, or third party software to integrate and maintain. [Joseph, 2006]

A Data Guard configuration consists of one active/production database and up to nine standby databases. In the scope of my thesis we will only use one standby database. A standby database is a consistent copy of the primary database. Data Guard does not use replication but transmits the redo logs to another database instance and thus keep active and standby database instances in sync. Redo logs are the logs which record altered data which have not yet been written to the data files. If the active database becomes unavailable, the standby database will be switched to become a new active database. Also a controlled switchover can be performed. [Bhanot and Kundu, 2006]

The databases in a Data Guard configuration are connected by Oracle Net and may be dispersed geographically in our solution. Data Guard is used for a site level failure protection, working perfectly with RAC. It provides low-cost redundancy since the requirement for the speed of connections is not so high. It does not require expensive and complex mirroring hardware or software either. [Pratt and Ray, 2005]

Oracle also has a product called Data Guard Broker, which is a distributed management framework that automates and centralizes the creation, maintenance, and monitoring of Data Guard configurations. The broker automates the previously manual process of configuration. Once we have created the Data Guard configuration, the broker monitors the activity, health, and availability for all systems in the Data Guard configuration. [Pradeep and Joudip, 2006] Unfortunately the Data Guard Broker is not supported with RAC.

Oracle provides the commands and configuration parameters to create a Data Guard configuration as well. As the Data Guard Broker is not supported with RAC in Oracle 9i, the creation of the configuration and the monitoring of the states of the database instances need to be implemented using the provided commands. Basically, configuring the Data Guard into use means tens of commands for both of the instances on each site, and the switchover/failover means a couple of commands for the surviving instances.

Data protection with Data Guard can be configured in three different modes to meet different goals. They are maximum protection, maximum availability and maximum performance. Maximum protection means that the data is transmitted to the standby synchronously, which affects the performance of the active database instance. Maximum availability configuration is the same as maximum protection with an exception, that when the standby becomes unavailable, the processing moves to asynchronous mode automatically. The risk of data loss is zero for both of them. Maximum performance

mode means asynchronous processing for the database transactions. It brings small risk of data loss usually at the level of zero up to few minutes.

It should be noted that with asynchronous processing the redo logs are transmitted to the standby nodes only after a log switch. This means that the standby is always a little bit behind the active instance for changes of the data. The log switch interval depends on the amount of “inserts” and “updates” in the database and on the sizes of the redo logs. It is typically from a couple of minutes to tens of minutes. The log switch can also be configured to happen after a certain time even if the log is not full yet.

5.4.2 Standby database

A standby database is a transactionally consistent copy of the primary database [Oracle, 2002]. A standby database is initially created from a backup copy of the primary database. Once created, Data Guard automatically maintains the standby database by transmitting the redo logs containing all the database data modifications from the primary database to the standby site and then applying the redo logs to the standby database. [Välimäki, 2005]

The standby database can be either physical or logical copy of the primary database. Both of them can offload the production database by performing the backup to tape or executing queries for reporting. The logical standby is not supported in our environment as some of the non-supported data types specified by Oracle are used in our application.

During a switchover operation in a Data Guard environment, there is no data loss, and the old primary database remains in the configuration as a standby database. During a failover (The starting of a service on another computer following failure on the primary computer.), some committed data might be lost depending on the data protection mode that is used.

5.4.3 Physical standby

A physical standby database is a fully identical copy of the active database in the database block level. (block-to-block copy)

A physical standby database has the following properties:

- The physical standby database is kept in sync with the primary database via applying redo logs block-by-block. It is referred as “Redo Apply”. [Joseph, 2006]

- The physical standby can be opened in “read-only” mode only when Oracle is not applying the redo logs. Traffic handling would intensively require “write” operation instead of “read-only”. In practise it means the standby database can not handle online traffic (calls) at all. So physical Oracle standby configuration can not be used if standby database is intended to handle online traffic. But some background statistics which only need “read” operation can be done when the replication is not happening at the same time.
- A physical standby database supports all data types, data definition language (DDL) and data manipulation language (DML) operations which are supported by primary database as well. It also provides safeguard against data corruptions and user errors. File system corruptions on the primary do not propagate to a standby because the received redo is validated before application to the standby. Similarly, logical corruptions or user errors that cause the primary database to be permanently damaged can be resolved. A standby can be configured to delay the application of changes received from the primary, thus a window to detect and prevent the propagation of errors from a primary to a standby is provided.
- The “redo applying” technology used by the physical standby database applies changes using low-level recovery mechanisms, which bypass all SQL level code layers and therefore is the most efficient mechanism for applying changes.
- With a physical standby database, the standby instance needs to be restarted in a switchover and failover. In switchover, the old active instance also needs to be restarted.

5.4.4 Logical standby

A logical standby contains the same data (rows) and structures as the active database has, but the structures are the same only on the logical level. Physical organization and structure can be different. A logical standby database provides similar disaster recovery, high availability, and data protection benefits as a physical standby database does. It is an open, independent and active database.

A logical standby database has the following properties:

- The logical standby database is kept synchronized with the primary database by transforming the data in the redo logs received from the primary database into SQL statements and then executing the SQL statements on the standby database. It is referred as “SQL Apply”. [Joseph, 2006]

- The logical standby is open and can also be used for read-only operations. Oracle prohibits the use of "write" operations on logical standby database.
- By using logical standby database, hardware resources on the standby site can be efficiently used. A logical standby database can be used for other business purposes in addition to disaster recovery requirements [Meeks, 2006a]. It can host additional databases schemas beyond the ones that are protected in a Data Guard configuration. Users can perform normal DDL or DML operations on those schemas at any time. Because the logical standby tables that are protected by Data Guard can be stored in a different physical layout from the primary database does, additional indexes and materialized views can be created to improve query performance and suit other specific business requirements. [Pradeep and Joudip, 2006] It is very useful that the standby can be accessed by other management application than the traffic handling application.
- Logical standby database reduces workload of primary database. A logical standby database can remain open at the same time while its tables are updated from the primary database via SQL. Those tables are simultaneously available for read access. This makes a logical standby database an excellent choice to execute queries, summations, and reporting activities, thereby the primary database can be offloaded from those tasks and saving valuable CPU and I/O cycles.
- With a logical standby database, the standby instances do not need to be restarted when a switchover or a failover happens.

Even though a logical standby database is more flexible than a physical standby database, it does not fit into our application. A logical standby database has some restrictions on data types, types of tables, types of data definition language (DDL) and data manipulation language (DML) operations. The following types are not supported in a logical standby database: NCLOB, LONG, LONG RAW, BFILE, ROWID, UROWID, user-defined types, object types REFs, varrays and nested tables. [Välimäki, 2005]

Unfortunately our application has the dependency on HP OpenCall platform. There are some uncertainties whether some of those restricted data types will be used in the future. Thus a logical standby database is not an option for us because we want to eliminate such limitations in applicability. [Välimäki, 2007]

5.4.5 Implementation details

From the comparison of the physical and logical standby features, we select the physical standby database in our solution. Implementing the site disaster tolerance solution by using physical standby database is decided based our key requirements. We consider the performance, availability, usefulness of read-only mode and cost.

The physical standby can be opened for read-only mode only when Oracle is not applying the redo logs. On the contrary the system is required to handle online traffic on both sites simultaneously. In order to solve this conflict, we will slightly modify the database structure.

The database needs to be divided into two and activated on each sites to handle traffic. In case of site disaster, we do not lose any traffic since calls can be rerouted to another site, which has an active database containing half subscribers, and an active replica containing the other half subscribers.

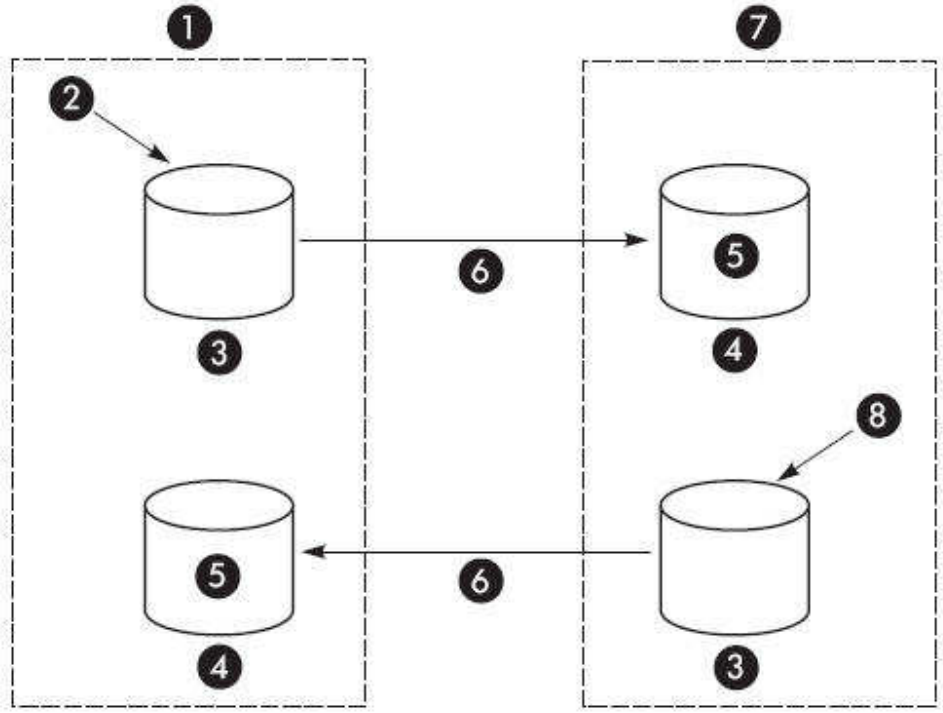

Figure 8. Two databases across two sites.

1. Site A.
2. Application writes to DB1, which has half subscribers, active on site A.
3. Source.
4. Destination.

5. Copy/replica, physical standby database.
6. Transfer direction.
7. Site B.
8. Application writes to DB2, which has another half subscribers, active on site B.

Namely we will have two databases DB1 and DB2. Each of them contains half subscribers and active at different sites. E.g. Primary/production database of DB1 is active at Site A and the standby/replica database of DB1 is read-only at Site B. They are the pair which are synchronized and under protection of Oracle Data Guard. On the other hand, primary/production database of DB2 is active at Site B and the standby/replica database of DB2 is read-only at Site A. Creation of the standby database needs our own implementation and instructions. Oracle Data Guard has to be set up on both database servers. The mount points and directories are named the same in both servers. The database name should be the same for the primary and the standby database.

The primary database polls its standby databases in every minute to see whether there is a gap in the sequence of archived redo logs.

If the application cannot get a connection to the database, it should wait long enough to know whether the reason is the database failover or not. Only after some time an alarm should be generated in case of failover.

The database is divided into two sites based on subscribers. On each site, there is only a information of one half of the subscribers available for call handling. Only the active database can serve the application at one site in normal situation. The standby database at the same site can only be used for statistic or backup purpose for another database, which is active at the other site. Only in a disaster situation, the two databases are active handling traffic at the same site, one of which is just activated and transitioned to the primary role taking over the data serving. There can be two ways to modify some applications to adapt this situation.

1. We do a small modification to an SCP application and add the connections between SCPs at the primary site and DB cluster nodes at the remote site. Whenever a request comes to SCP, the instance submits the request to the local primary database in the DB cluster. In case the subscriber can not be found in the local primary database, the instance does not return an error immediately. Instead, the instance forwards the request to the primary database in the DB cluster at the remote site, which holds the other half subscribers' information. This solution will

prolong the call handling time and lower the performance seriously because of the increasing traffic load between two sites. It is not suggested as an efficient solution. [Niemi, 2007]

2. We create and maintain the database location of a subscriber in Home Location Register (HLR) [Niemi, 2007]. The Home Location Register (HLR) is the main database of permanent subscriber information for a mobile network. In the message of Originating CAMEL Subscription Information (O-CSI), the HLR informs a Service Switching Point (SSP) which SCP it should connect to by the "gsmSCF address" parameter [3GPP, 2006]. This "gsmSCF address" indicates the location of an SCP which connects to the primary database containing this specific subscriber. This solution increases the performance at the cost of increasing management workload. Whenever there is a new subscriber added into the local primary database in the DB cluster, the identification of that subscriber must be associated to the list in the HLR which serves this subscriber. For this purpose, one trigger in the local primary/active database must be created to inform a separate application to update the list in the HLR whenever there is a new subscriber added. These updates can be performed instaneously or with some delay. Thus the HLR maintains the newest subscriber list for query. The discussion of such an interface towards the HLR is not covered in this thesis.

Of course there are more than the above two ways to modify other applications in order to adopt this replication solution. The detailed-level configuration and implementation of SCP/HLR connections are out of the scope of this study. This thesis concentrates on the database replication solution.

5.4.6 Availability

If the primary database becomes unavailable (disasters, maintenance), the standby database can be activated and can take over the data serving needs of the SCP. SCP will reroute traffics to the working site. If the standby database node fails it must not affect the functionality of the active one. In case the standby database node fails it must be booted and recreated automatically. An alarm will be created if the new standby database cannot be recreated with one attempt.

There can be several standby nodes and thus the availability of data is generally good. But in this thesis, only one standby is presented.

All in all, Data Guard and RAC are complementary and should be used together. RAC provides high availability at one site, while Data Guard adds value to disaster protection at the same time. RAC provides rapid and automatic recovery from node failures or an instance crash at one site. Oracle Data Guard protects against site disasters, data corruption and user errors by maintaining transactionally consistent copies of primary site.

5.4.6.1 Failover

Failover happens automatically to the standby database node if the active database node fails. Starting with Oracle Database 10g Release 2, Data Guard configurations using Maximum Availability mode have the option of implementing unattended, automatic failover upon primary database failures [Meeks, 2006b]. Graceful failover and forced failover are the possibilities for a failover with Oracle Data Guard. Graceful failover is preferred and only if it does not succeed, forced failover is tried.

If the standby site that is transitioning into the role of primary site contains a physical standby database, then the instance will be restarted after the failover operation completes. If the site contains a logical standby database, the instance does not need to be restarted.

The original primary database is removed from the Data Guard configuration after failover. Recovery of the failed database node can happen automatically.

5.4.6.2 Switchover

During a switchover operation, there is no data loss, and the old primary database remains in the configuration as a standby database. A switchover is typically used to reduce primary database downtime during planned outages, such as operating system or hardware upgrades. A switchover operation takes place in two phases. In the first phase, the existing primary database is transitioned to a standby role. In the second phase, the old standby database is transitioned to the primary role. [Välimäki, 2005]

No committed data would be lost during the database switchover. Before the switchover it must be verified that there are no active users connected to the databases.

If the switchover operation transitions a physical standby site to the primary role, then both the primary database and the target standby database will be restarted after the switchover operation completes.

If the switchover operation transitions a logical standby site to the primary role, nothing needs to be restarted after the switchover operation completes. Neither the primary database nor the logical standby databases need to be restarted.

5.4.7 Performance

The more up to date the standby node needs to be the more it has implications to the active node's performance. Definitely data protection mode has impact to performance. We must balance cost, availability, performance, and transaction protection.

5.4.7.1 Scalability

The active / standby configuration by Data Guard can be scaled up by adding resources to the existing nodes and thus there are limits to the scalability.

5.4.7.2 Load balancing

The standby database can field queries and run backups to relieve processing demands on the primary database. It can accept the background statistic tasks when the redo log is not applying. It can not handle the real time query because of the synchronization from primary database does not allow the read operation at the same time. In order to prepare for the disaster, the traffic load is 50% of the capacity on each site when they are healthy.

5.4.7.3 Bandwidth and Latency

Even though a write transaction affects the redo log files, archive log files and data files, Data Guard sends only the redo data to keep the standby databases synchronized with the primary. This is in contrast to certain storage-level remote mirroring solutions which may send all of those changes, requiring up to 3 times more network resource consumption. Remote-mirroring solutions based on storage systems often have a distance limitation due to the underlying communication technology (Fiber Channel, ESCON) used by the storage systems. In a typical example, the maximum distance between these two boxes connected in a point-to-point fashion and running synchronously can be only 10 km. By using specialized devices this can be extended to 66 km. However, when the standby data center is more than 66 km apart, a series of repeaters and converters from third party vendors have to be used. These devices convert ESCON/Fiber Channel to the appropriate IP, ATM or SONET networks. [Oracle, 2006]

The formula [Dutcher and Ray, 2006] shows the correlation between the redo rate and the required bandwidth (assuming a conservative TCP/IP network overhead of 30%) is:

Required bandwidth = ((Redo rate bytes per sec. / 0.7) * 8) / 1,000,000 = bandwidth in Mbps.

The pilot project in Fannie Mae [Oracle, 2007] showed Oracle Data Guard could sustain synchronous zero data loss protection and avoid any loss of information for workloads running as high as 16 megabytes per second of redo data. It could run at even faster rates in asynchronous mode for applications that do not have such stringent data loss requirement.

For Maximum Protection and Maximum Availability, the redo data transport setting requires synchronous mode. For the Maximum Performance mode, the redo transport is set to the asynchronous mode. This implies that a possible impact on the production transactions is correlated to the latency of the network link between the primary and the standby. Since the latency for a network is usually correlated to the length of the network, Maximum Protection and Maximum Availability modes are not recommended for Data Guard deployments over a Wide Area Network (WAN). As we all understand that WAN links are generally more expensive, less reliable, have lower bandwidth and higher latency than LAN links. For WAN deployments of Data Guard, the Maximum Performance protection mode is recommended. All three protection modes can however be used for Local Area Network (LAN) or Metropolitan Area Network (MAN) deployments of Data Guard. [Ray and Ashish, 2006]

5.4.8 Client connections in failovers

It is possible to configure Oracle to hide some of the system failures from database client connections.

It is important to hide system failures from database client connections. Such connections can include application users in client server environments or middle-tier database clients in multi-tiered application environments. In our system, SCP application is connected to database via ExDat application which is a middle-tier client. Properly configured failover mechanisms transparently reroute client sessions to an available node in the cluster. This capability in the Oracle database is referred to as Transparent Application Failover (TAF). The new database connection which is achieved using a different node is identical to the original, while active transactions roll back. This is possible regardless of how the connection fails. [Välimäki, 2005]

TAF refers to the failover and reestablishment of application-to-service connections. It enables client applications to automatically reconnect to the database if the connection

fails, and optionally resume a `SELECT` statement that was in progress. This reconnect happens automatically within the Oracle Call Interface (OCI) library. [Oracle, 2005]

In order to use this feature, the application needs to use failover-aware API calls from the Oracle Call Interface (OCI).

Transactions involving `INSERT` `UPDATE` or `DELETE` statements are not supported by TAF. Only `SELECT` failover or `SESSION` failover are supported. [Välimäki, 2005]

6 Conclusion

In this thesis I studied how to synchronize data in a replicated distributed database environment. In order to enhance the disaster tolerant feature, another data center is introduced.

Effective high availability architecture should keep data and system protected against outage, while also contributing processing power for daily tasks when crises are not looming. Thus the two sites should be active simultaneously handling traffic. In case of disaster, the entire site activities (traffic, provisioning, etc.) switchover needs to be as fast as possible, with minimal data loss. The data must be synchronized between two sites. Finding a less expensive, more reliable, high bandwidth and low latency synchronization solution is our target.

The major decisions to be made in designing a successful data synchronization solution in a replicated distributed database environment are explained in this study. The analysis is validated based on the experimental data from our partners.

One design consideration is choosing the replication mode. If expensive high-speed connections between two sites can be accepted and there is a stable network environment, the synchronous replication mode should be used to avoid the replication conflicts and provide zero data loss. Asynchronous mode is useful for applications which can tolerance some data loss to achieve higher level of performance.

Another design consideration is the distance between two sites. There are some inherent limitations in some transmission technologies preventing us to adopt such technologies. RAC on HP extended cluster is a good example. It is a simple and effective option but accompanying with the shortage of 100 kilometer limitation for the inter-site distance.

The third design consideration is cost. In addition to showing how distance and bandwidth affect performance and cost, optional configurations and key planning for different transmission technologies are described in the thesis. If only low-speed

connection can be afforded, standby database is an ideal option to keep data synchronized between two sites. It propagates only changes in stead of every I/O “writes” of a primary database to a standby database with less expensive connection, whereas RAC on HP extended cluster requires expensive DWDM connection.

There are two logical databases at two sites for either array-based or software-based replication solution from management point of view. Data exist at both sites and synchronized with each other. Both databases are active and able to handle traffic. In case we use asynchronous mode to synchronize data, there is replication conflict to be solved. The recovery of synchronization can not happen automatically without human interference. Both solutions can achieve high performance if configured properly. The difference is the effort of deployment. Database files, the redo log files, and the control files must be replicated in the array-based replication solution resulting in heavy traffic load between two sites and deep knowledge to plan tables on virtual disks, while only redo log files needed to be transmitted over network in software-based solution. Since array-based solution synchronize almost every I/O “writes” happened at one site, it can be used to protect the non-database data as well.

RAC on HP extended cluster solution implement a single RAC database spanning two geographically distant sites. The physical location of data in storage and replications are transparently managed by RAC. RAC can also automatically balance transaction processing among cluster nodes across sites.

I studied effectiveness of above existing solutions, and presented a theoretical foundation for data synchronization over larger distances in relatively cheap standard networks [Dutcher, 2006]. This new solution is to integrate a physical standby database managed by Oracle Data Guard. Each site has two databases. The primary database containing half subscribers’ data is active to handle traffic. The standby database is just a copy of the other primary database containing the rest half subscribers’ data at the remote site. It can not handle the traffic with the read-only limitation. So there are altogether four databases across two sites. A primary database and its standby database are a pair under configuration and protection of Oracle Data Guard. The data replication and consistency are guaranteed by applying redo logs from a primary database to its standby database at the remote site. There is no replication conflict because the same subscriber data is not updatable simultaneously at two sites. This solution transfer only redo logs without expensive and complex HW/SW mirroring, thus improves performance and save cost significantly. Besides, standby database does more than providing protection from disaster, e.g. providing user error tolerance by logically validating the changes before

applying them on standby database. This solution provides a chance to win huge business success by staying within budget, getting the relatively high performance and availability, deriving the most out of disaster recovery infrastructure (even in times when there is no disaster) [Baird, 2004].

Since the capability of HW and workload can influence the performance significantly, this proposed solution would require further validation with actual traffic for our application and network environment, and of course it requires more work to be done in the functional testing and performance testing. It is very hard to get the experimental data from our partners for exactly the same configured HW and network traffic as in our network. Nevertheless, tuning the system and adjustment of configuration are next steps to achieve the best performance under the practical system load and connection technologies, and therefore they could be interesting for a further study with the mind of getting better network efficiency, higher flexibility and better data resilience and convergence across two sites.

Reference

[Ahonen, 2003] Jukka Ahonen, *Stored Procedures for the External IN Database,* internal document, Dec, 2003.

[Ahonen, 2004a] Jukka Ahonen, *Nokia nCreate Platform J5 Architecture*, internal document, Oct, 2004.

[Ahonen, 2004b] Jukka Ahonen, *External IN database Implementation Specification*, internal document, Jan, 2004.

[Bauer, 2002a] Mark Bauer, *Oracle9i Real Application Clusters, Concepts, Release 2(9.2), Mar, 2002.*

[Bauer, 2002b] Mark Bauer, *Oracle9i Real Application Clusters, Administration, Release 2(9.2), Mar, 2002*

[Baird, 2004] Cathy Baird, *Oracle Database, High Availability Architecture and Best Practices, 10g Release 1 (10.1),* June, 2004.

[Bhanot and Kundu, 2006] Pradeep Bhanot and Joydip Kundu, *Using Oracle Data Guard to Protect Your Enterprise Database,* internal technical proposal, 2006.

[Burroughs, 2002] Ted Burroughs, *Oracle 9i Advanced Replication, Release 2 (9.2),* Mar, 2002.

[Cutler et al., 2005] Mai Cutler, Sandy Gruver and Stefan Pommerenk, *Extended Distance RAC, Eliminating the current physical restriction of Oracle Real Application Cluster*, 2005.

[Cuttaz, 2006**]** Stephane Cuttaz, *NOKIA Disaster Recovery Plan*, internal technical proposal, May, 2006.

[Dutcher, 2006] Ray Dutcher, *Oracle9*i *Data Guard: Primary Site and Network Configuration Best Practices, white paper,* June, 2006.

[Dutcher and Ray, 2006] Ray Dutcher and Ashish Ray, *Network Bandwidth Implications of Oracle Data Guard,* 2006.

[HP, 2004], *HP Extended Cluster for RAC - 100 kilometer separation becomes a reality*, April, 2004.

[HP, 2005a] S*torage virtualization and the HP StorageWorks Enterprise Virtual Array*, 2005.

[HP, 2005b] *HP-UX GEOGRAPHICALLY DISPERSED CLUSTERS,* 2005

[HP, 2006] *HP StorageWorks Continuous Access EVA planning guide,* July 2006.

[Kemppinen, 2004] Simo Kemppinen, *J5 Database Description*, internal document, Nov, 2004.

[Laurila, 2005] Eero Laurila, *J5 SMP RAC Level 1+2 Backup and Recovery, Implementation Specification,* July, 2005.

[McElroy and Pratt, 2005] Patricia McElroy, Maria Pratt, *Oracle Database 10g: Oracle Stream Replication, white paper,* May, 2005.

[McElroy et al., 2005] Patricia McElroy, Maria Pratt, Bob Thome, *Sharing Information with Oracle streams, white paper,* May, 2005.

[Meeks, 2006a]Joseph Meeks, *Data Protection, The Oracle Data Guard Difference - It's Not Just About Disasters,* 2006.

[Meeks, 2006b]Joseph Meeks, *Implementing Data Guard- What you need to know?* 2006.

[Niemi, 2007] Kari Niemi, internal oral discussion, Feb, 2007.

[Nokia SQL script, 2004] *Prepaid data package body,* SQL script, May, 2004.

[Oracle, 2002] *Oracle Data Guard Concepts and Administration manual, Release 2 (9.2)*, Oct, 2002.

[Oracle, 2004] Oracle Magazine, March, 2004, http://otn.oracle.com/oraclemagazine. Checked January 26, 2007.

[Oracle, 2005]Oracle® Application Server High Availability Guide, 10g Release 2 (10.1.2) http://download-east.oracle.com/docs/cd/B14099_11/core.1012/b14003/title.htm Checked January 26, 2007.

[Oracle, 2006] Oracle Data Guard and Remote Mirroring Solutions, 2006. http://www.oracle.com/technology/deploy/availability/htdocs/DataGuardRemoteMirroring.html. Checked January 26, 2007.

[Oracle, 2007] Oracle Magazine, January, 2007, http://otn.oracle.com/oraclemagazine. Checked January 26, 2007.

[Peterson, 2006] Erik Peterson, *RAC on Extended Distance Clusters*, 2006.

[Pratt, 2001] Maria Pratt, *Oracle 9i Replication*, White paper, June, 2001.

[Pratt and Ray, 2005] Lyn Pratt, Ashish Ray, *Best Practices for Achieving Business Continuity using Oracle Applications and Oracle Database Technology, white paper*, Sep, 2005.

[Slee, 2004] Roland Slee, *Oracle 10g RAC Value Proposition*, Oct, 2004.

[Seikku, 2004] Ermo Seikku, HP *Enterprise Virtual Array Technical overview*, Power point presentation, Feb, 2004,

[Soppi, 2004] Rauno Soppi, *External IN database Implementation Design*, Jan, 2004.

[To and Meeks, 2006] Lawrence To and Joseph Meeks, Oracle Database 10g Release 2: Roadmap to Maximum Availability architecture, white paper, Apr, 2006.

[Urbano, 2003a] Randy Urbano, *Oracle Database Advance Replication 10g Release 1*, Part No B10732-01, Dec, 2003.

[Urbano, 2003b] Randy Urbano, *Oracle Streams, Replication Administrator's Guide, 10g Release 1 (10.1)*, Dec, 2003.

[Välimäki, 2005] Johanna Välimäki, *Database high availability with Oracle Data Guard*, internal paper, Nov, 2005.

[Välimäki, 2007] Johanna Välimäki, Internal oral discussion, Feb, 2007.

[Weygant, 2001] Peter S. Weygant, *Clusters for High Availability, A Primer of HP solutions*, Second Edition, Prentice-Hall, 2001.

[Wikipedia] Wikipedia: The free encyclopedia. http://en.wikipedia.org/wiki/Intelligent_network. Checked January 26, 2007.

[Özsu and Valduriez, 1999] M. Tamer Özsu, Patrick Valduriez, *Principles of Distributed Database System*, Prentice-Hall, 1999.

[3GPP, 2006] 3GPP TS 23.008 V7.4.0, Technical Specification, Dec, 2006.