Fall 2009

Database System Architecture for Fault tolerance and Disaster Recovery

Anthony Nguyen
Regis University

Regis University
College for Professional Studies Graduate Programs
Final Project/Thesis

Disclaimer

Abstract

Application systems being used today rely heavily on the availability of the database system. Disruption of database system can be damaging and catastrophic to the organization that depends on the availability of the database system for its business and service operations. To ensure business continuity under foreseeable and unforeseeable man-made or natural disasters, the database system has to be designed and built with fault tolerance and disaster recovery capabilities. This project explored existing technologies and solutions to design, build, and implement database system architecture for fault tolerance and disaster recovery using Oracle database software products. The project goal was to implement database system architecture for migrating multiple web applications and databases onto a consolidated system architecture providing high availability database application systems.

Acknowledgements

I would like to thank my project advisor Ms. Shari Plantz-Masters for her time to review and provide invaluable comments to help me get my best out on the paper content and organization. I would also like to thank Professor Donald Ina for his guidance, direction, and encouragement to get the project paper to completion.

I would like to express my deep gratitude to my wife and children for their love and support for my continuing education and other endeavors. Without them, I would not have the energy and enthusiasm to set and reach worthy goals.

Table of Contents

Abstract ii

Acknowledgements iii

Table of Contents iv

List of Figures vii

List of Tables viii

Chapter 1 – Introduction 1

1.1 Problem Statement 1

1.2 Review of Existing Situation 2

1.3 Goals of the Project 3

1.4 Issues and Limitations 3

1.5 Scope of Project 4

Chapter 2 – Review of Literature and Research 5

2.1 High Availability Characteristics 5

2.2 High Availability Requirements 6

2.2.1 Requirements Analysis 7

2.2.2 Risk Analysis 9

2.2.3 Cost Analysis 11

2.3 High Availability Functions and Capabilities 15

2.4 High Availability Existing Solutions 16

2.5 Research Methods 20

2.6 Research Summary 21

Chapter 3 – Project Methodology 23

3.1 Planning 23

3.2 Analysis.....24

3.3 Design 24

3.4 Implementation 25

Chapter 4 – Database System Architecture for Fault Tolerance and Disaster Recovery ..27

4.1 Database System Architecture Design.....27

4.2 Oracle Database System Architecture Design 31

4.3 Implementation Plan 37

4.3.1 Overview.....38

4.3.2 Scope.....39

4.3.3 Current System Architecture.....40

4.3.4 Proposed System Architecture.....41

4.3.5 Database Migration and Upgrade Test Plan 42

4.3.6 System Architecture Test Plan.....43

4.3.7 System Cut-over.....44

4.3.8 Timeline 44

4.4 Implementation Procedures 45

4.4.1 Oracle 10g RAC Software Installation 45

4.4.2 Oracle Database Upgrade from 9i to 10g 48

Chapter 5 – Project History.....51

5.1 How the project began 51

5.2 Changes to the project plan.....51

5.3 What went right and what went wrong 52

5.4 Project variables and their impact 52

5.5 Results summary 53

Chapter 6 – Lesson Learned 54

6.1 What was learned from the project experience 54

6.2 Did project meet initial project expectations 54

6.3 Next stage of project if it continued 54

6.4 Conclusions/recommendations 55

References 56

Appendix A - System Requirements Questionnaire……..…….…….....568

List of Figures

Figure 1: Most Common Causes of Unplanned Downtime ……..…….…. 10

Figure 2: Acceptable downtime measurement ……...… 13

Figure 3: Systems Development Life Cycle (SDLC) ……...…..……. 23

Figure 4: High Availability Database System Architecture ……...… 30

Figure 5: High Availability Oracle Database System Architecture ……..…. 34

Figure 6: Oracle Standby Database Implementation …….…….…… 36

Figure 7: Current System Architecture …….……..…….…….……. 40

Figure 8: Proposed System Architecture …….……...…….…….. 41

Figure 9: System Implementation Timeline …….…….. 44

List of Tables

Table 1: Disaster Recovery (DR) Service Levels ……..……..…… 8

Table 2: Acceptable downtime measurement ……..…….. 12

Table 3: The Direct Costs of Downtime ….……..…….. 14

Table 4: Microsoft SQL Server Disaster Recovery Options …….……..… 28

Table 5: Oracle Database High Availability Architecture …….……32

Chapter 1 – Introduction

1.1 Problem Statement

This project involved the design and implementation of new database system architecture for a program office within the Department of Justice (DOJ) to consolidate three separate database systems supporting three web applications onto one consolidated database system architecture. The DOJ web applications are used for the collection, dissemination, and collaboration of law enforcement investigation activities. These applications were developed and implemented many years ago at the early stages of web technology and had been continually upgraded over the years. However, other than software version upgrades, the system architectures to support these applications basically remained unchanged. The applications were developed in Java and the databases were running on Oracle. Each database supported its own application. Each application had its own system architecture. The three system architectures were very similar but independent of one another.

The main problem was that for each application, the application server and database server were implemented on the same physical server with no redundant or backup server. Even though the servers were all Sun servers, they were different models and had different hardware configurations.

Another problem was the application and database servers were running different software versions. The non-uniformity in the software and hardware that were being used for each application and database system made it hard to maintain the systems, to trouble shoot and resolve problems when they occur. Furthermore, the problem

resolutions could not be applied across different application systems because of their different configurations. Each application and database system had to be maintained separately since they were running on separate servers.

1.2 Review of Existing Situation

The DOJ applications were built on three-tier system architecture. The first tier was the client web browser to log on to the application. The second tier was the application server that contained and executed application programs. The third tier was the database server that processed data and queries coming from the application. When the systems were built, the application and database were put on the same server. This setup had worked for several years because there were not so many transactions being handled by the applications and the user community at the time was rather small. As the number of users and the transaction volume grew significantly over the years, system performance was severely degraded.

For each system, the application and database were implemented on a single server with no system redundancy. When one of the systems failed beyond repair, the disaster recovery plan relied on using one of the systems that had spare operational capacity as a new host to rebuild the failed system. The application (second tier) and the database (third tier) should have been implemented on separate physical servers for better system performance, fault tolerance, and ease of maintenance.

The applications were developed in Java running on Tomcat web servers implemented on Sun Solaris 9 operating system. The three databases were set up on three separate servers using Oracle 9i database software on Sun Solaris 9 operating system. Oracle 9i and Solaris 9 were at the end of their support cycle. As the systems

were being redesigned, the DOJ program office also planned to upgrade the Oracle databases to Oracle 10g and Solaris 9 to Solaris 10 to stay current with advanced technology.

1.3 Goals of the Project

The main objectives of the project are to reduce the number of database systems and to simplify database system operations and maintenance tasks while ensuring high system availability for all applications.

One of the goals was to find solutions provided by commercial hardware and software vendors and recommend approaches to design and build reliable database system for fault tolerance and disaster recovery.

Another goal was to design and implement a consolidated database and application system architecture to migrate existing application and database systems to the consolidated system architecture, to enhance system fault tolerance and disaster recovery capability, and to ease the maintenance tasks by reducing the number of servers to be maintained.

1.4 Issues and Limitations

The DOJ program office was using Oracle database software products for its databases and Sun Solaris operating system for its servers and would like to stay with Oracle and Sun products because of the financial and technological investments that it had put into its existing application systems. The program office had interests in the research for technologies offered by other hardware and software vendors that could be considered to build database system architecture for fault tolerance and disaster recovery

for future planning, but heavily leaned toward Oracle and Sun products to ease the learning curve for its technical staff and to speed up the implementation timeline.

1.5 Scope of Project

This project concentrated on the study of the characteristics of a high availability system as well as the technologies that could be applied to design and build database system architecture for fault tolerance and disaster recovery. The project geared toward the design of a consolidated database system architecture that could be implemented to migrate and upgrade existing database application systems for the DOJ program office. The database system architecture that was designed and implemented in this project incorporates common features of commercially available database software products so that the database system architecture design can be applied to different database software products and independent of the database software products.

Chapter 2 – Review of Literature and Research

2.1 High Availability Characteristics

Database system architecture that provides fault tolerance and disaster recovery capabilities is often specified as high availability database system. High availability refers to the characteristics of a system that allow the system to sustain continuous operation in the event of hardware and software failures due to natural or man made causes (Webb, 2008). A highly available system usually survives failures by substituting standby hardware or software components to reproduce normal functions to withstand system disruptions. A high-availability system can also allow replacement of failed components or perform system upgrade without disrupting system operations.

The availability of system is normally defined as (Marcus, 2003):

$$A = \frac{MTBF}{MTBF + MTTR}$$

Where A is system availability, MTBF is the mean time between failures, and MTTR is the mean time to recover the system. From the formula for system availability, it can be derived that when MTTR approaches zero (i.e., system down time is substantially short), availability (A) increases toward 100 percent. On the other hand, when MTBF gets larger (i.e., system down time occurs very rarely), MTTR has less impact on A. Therefore, the goal of system high availability is to make MTTR as small as possible and MTBF as large as possible.

System outages and downtime are considered inevitable. However, the downtime can be reduced by taking steps to minimize the duration of the system outages. Based on the formula for system availability shown above, there are two ways to improve system availability: either extend MTBF (keep the systems' components from failing) or extend MTTR (shorten the recovery time when they do fail). It may be impossible to predict when a component will fail, but the system can be designed to be protected against component failures and to reduce the amount of time to repair or replace the failed components. For example, by implementing disk mirroring, system downtime can be reduced from hours to no down time for problems caused by a failed disk; by setting up clustering and system failover, system downtime can be reduced from days to just a few minutes for system hardware failure.

2.2 High Availability Requirements

The high availability capabilities of a system are defined by the system requirements analysis. The main objective of the requirements analysis is to determine the business needs and identify possible disasters that can happen to a database system. The system requirements analysis essentially analyzes capabilities that the database system should have to serve business needs and identifies the risks that may affect database system operations so that preventative measures can be planned, developed, and implemented to eliminate the risks as much as possible. The more preventative measures that are built into the database system, the lower the chance that the system operations will be disrupted or damaged such that the database cannot be recovered when a disaster strikes. On the other hand, no matter how sophisticated the database system is to be built with all of the preventative measures, there is always residual risk of system disruption

and outage. The system requirements analysis consists of the following areas: requirements analysis, risk analysis, and cost analysis.

2.2.1 Requirements Analysis

The purpose of requirements analysis for database fault tolerance and disaster recovery is to determine the timeframe in which the database system can be recovered after a disaster so that appropriate solutions can be utilized in building the system. Sun BluePrints OnLine specifies the following classification levels of disaster recovery capability that can be used as a guideline for implementing high availability system architecture (Stringfellow, 2000).

Table 1: Disaster Recovery (DR) Service Levels

Disaster Recovery Classification	**Time to Recover**	**Acceptable Data Loss (from Time of Failure)**	**Typical Implementation**
AAA	Four Hours or Less	Maximum one hour	Database Replication and/or Network Mirroring
AA	Four to 12 Hours	Four hours	Standby Database
A	12 to 24 Hours	24 hours	Usually restore from offsite backup
B	24 to 72 Hours	24 hours	Always restore from offsite backup

The table above indicates that the less time it takes to recover the system would lower system down time and that in turn ensures higher system availability. It also recommends the technologies that can be implemented to achieve system high availability.

2.2.2 Risk Analysis

A system requirements analysis may include a risk analysis that deals with the threats to the system. The threats may be classified as natural, technical, or human threats. The following are some of the threats that must be planned for and dealt with to ensure system high availability. These threats when they occur at the very least could disrupt system operations or in more severe situations could totally break the system (Wold, 1997):

Natural Threats: Fire, tornado, hurricane, and flood are calamities that can destroy the building facility where the system is located or damage the equipment that the system depends on.

Technical Threats: Power failure/fluctuation, malfunction or failure of CPU, system software, application software, network communications, heating, ventilation or air conditioning would affect system performance and bring the system down.

Human Threats: Computer virus, computer crime, burglary, vandalism, terrorism, civil disorder, sabotage, explosion, and war can cause undetected yet irreparable damages to the system.

The following figure shows the results of a survey conducted by computer industry analysts from Gartner/Dataquest on system risks that caused system downtime for real time systems (Marcus, 2003).

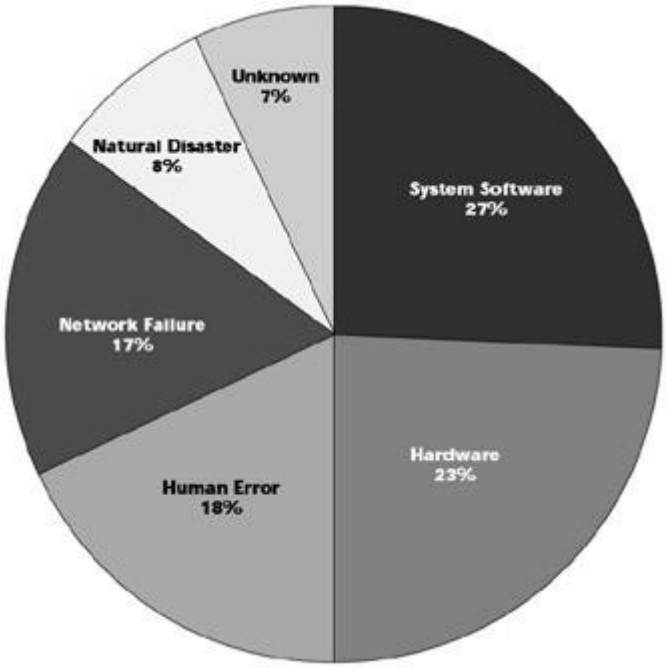

Figure 1: Most Common Causes of Unplanned Downtime

According to the chart, the greatest causes of system downtime are often contributed by software failures, hardware failures, human error, and network failures. These common causes together accounts for up to 85% of system downtime in which the hardware and software failures are the most common. Based on the chart above, system high availability can be greatly increased by reducing the chance of system hardware, software, or network failures and by circumventing human errors in managing the system as much as possible. A high availability system must have good system architecture with system redundancy to withstand the failures of system components. Additionally it must be managed by highly qualified operations staff with well established procedures put in

place to prevent errors from happening. As a result of the risk analysis, a disaster recovery plan should be developed to specify procedures to handle the threats to the system and to provide instructions to repair and recover the system to minimize system downtime when the disasters occur.

2.2.3 Cost Analysis

A cost analysis should be included in the system requirements analysis to weight the benefits against the costs of building high availability system architecture in order to fine tune the system requirements. The cost of building a high availability database system is determined by the recoverability and availability of the system.

Burleson (2005) defines recoverability as the ability of a system to be recoverable from failures with minimal downtime. Database recoverability is defined as the amount of time it would take to bring a database system back up and running if the system crashes due to reasons such as power surge, power failure, network failure, CPU failure, etc. Database recoverability directly relates to the effectiveness of the database backup and disaster recovery strategy. Ideally, the system should be up and running as quick as possible after a disaster. However, database size and the interval at which database backups are performed significantly affect the recovery time for a database.

Choosing a good backup scheme also depends on the recovery time allowed, or the mean time to recover (MTTR). MTTR is the desired time required to perform instance or media recovery on the database. A variety of factors influence MTTR for media recovery, including the speed of detection, the type of method used to perform media recovery and the size of the database. MTTR of a system should be very low.

Availability is measured by the amount of time the system has been up and is available for operation. In defining availability of the system, the word 'system' does not apply to just the database tier or the application tier, but to the complete enterprise system. This implies that every piece of equipment, including networks, servers, application controllers, disk subsystems, etc., should be considered for availability. Making all tiers of the enterprise system available also means that each tier should provide redundant hardware, helping to provide continuous service when one of the components fails.

The level of system availability is based on operational requirements of the system. If the requirement is 99.99% uptime in 24 hours a day for 365 days of the year, redundant architecture often becomes a necessary to ensure system accessibility at all times and to maintain system transparency in system fail-over scenarios. If some amount of downtime is allowed that does not affect the entire business, some of the redundancy may not be required.

The following table shows system availability requirement as measured by the amount of downtime allowed per year (Burleson, 2005).

Table 2: Acceptable downtime measurement

Availability Requirement	**Expected Downtime per Year**
99.995%	0.5 hours
99.97%	2.5 hours
99.8%	17.5 hours
99.5%	43.2 hours or 1.8 days
99%	88.8 hours or 3.7 days
98%	175.2 hours or 7.3 days
96%	350.4 hours or 14.6 days

Table 2 provides the expected downtime per year for the various levels of system availability requirements. The table illustrates the fact that the cost of availability rises substantially with each fraction increase in the availability requirement. Table 2 can be graphically represented in Figure 2 as follows (Burleson, 2005):

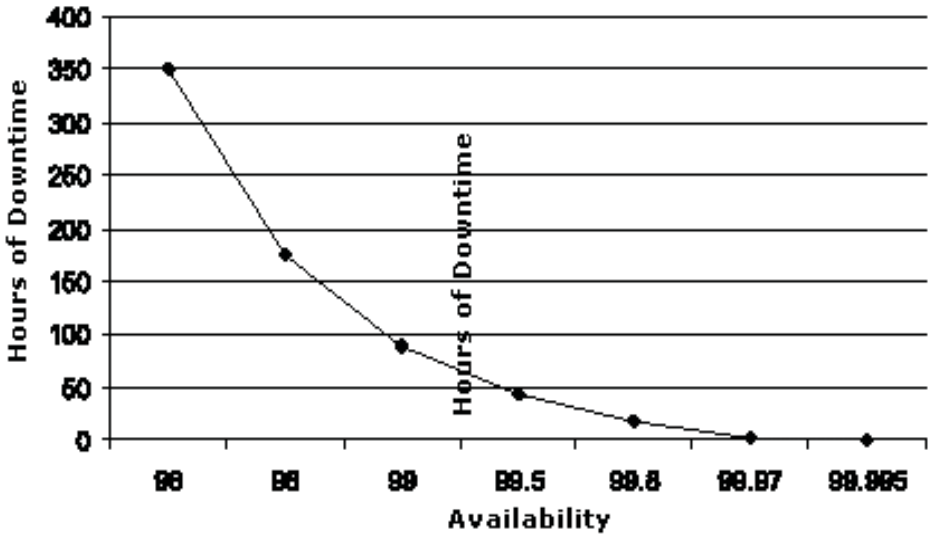

Figure 2: Acceptable downtime measurement

The cost of system downtime can be direct or indirect. The direct costs can often be put in dollar figures, while the indirect costs are much harder to calculate.

- Direct Costs of Downtime

Direct costs of downtime are mainly loss of productivity. The following table provides samples of direct costs of downtime per hour as reported by various services and industries (Marcus, 2003):

Table 3: The Direct Costs of Downtime

Industry	Average downtime cost per hour
Brokerage services	$6.48 million
Energy	$2.8 million
Credit card	$2.58 million
Telecomm	$2 million
Financial	$1.5 million
Manufacturing	$1.6 million
Financial institutions	$1.4 million
Retail	$1.1 million
Pharmaceutical	$1.0 million
Chemicals	$704,000
Health care	$636,000
Media	$340,000
Airline reservations	$90,000

- Indirect Costs of Downtime

Though the direct costs of downtime can be expensive, the indirect costs can be significantly higher and have much greater long-term impact to the organization. The following are examples of indirect costs: When the web site of a business organization does not function that prevents the customers from making online purchases or performing other business transactions, the customer would get frustrated. That experience would cause customer dissatisfaction and the customers may never buy products or do business with the company again. If the system outage becomes the news (especially for a well-known business organization) it may be a bad publicity incident. Consequently, the bad publicity could lead to company stock price losing value. There could be legal liability from lawsuits brought by stock holders because of the unexpected drop in the stock price. When things are going so bad in the company, employees get

discouraged and start to leave the company and that certainly would affect employee morale. The reputation of the company could be significantly damaged when the bad situation gets out of control or goes on over a long period of time.

2.3 High Availability Functions and Capabilities

A high availability system must at least have the following functions and capabilities (Marcus, 2003):

- Backups and Restores
- Highly Available Data Management (RAID, data redundancy, disk space and file system management)
- Clustering and Failover
- Data Replication
- Data Storage Technology (SAN – Storage Area Network, NAS – Network Attached Storage)

It is noted that with the implementation clusters and data storage technology, a high availability system can also be considered as a high performance computing system (Boukerche, 2007).

2.4 High Availability Existing Solutions

A database system built for fault tolerance normally has good system backup and recovery plans with redundant hardware/software to prevent loss of data or to enable the recovery of the loss data. Database disaster recovery plans vary greatly and often involve minimal to complete data recovery and restoration. The data recovery and restoration process itself may be based on time-consuming manual recovery methods or fully automated recovery mechanism. Solutions selected for database disaster recovery plan are often determined by the desired level of system availability and the acceptable data loss in the event of system outage occur. The following are the questions that can be asked to help define database system fault tolerance and disaster recovery objectives. These questions are aimed to quantify the acceptable system downtime, the allowable amount of data that might be lost, whether it is possible to recover the lost data, the time to recover lost data, and if there is a good chance to bring the system back on. How long can an organization or business afford to be without critical data and applications? How much data can it afford to lose? What is the likelihood that the data will be corrupted when system failure occurs? Can the data be recreated after system failure? If a network goes down, how long will it take to restore it? How will that affect the ability to conduct business? If the entire site where the system is physically located suddenly becomes disrupted and out of service, how will that impact organization or business operations? Are there other alternate sites available to be put in service? The answers to these questions would help define system requirements and the level of high availability that the system should be planned and built for.

To classify database system fault tolerance and disaster recovery capabilities, IBM and its user group have jointly developed seven tiers of system disaster recovery classification to define disaster recovery levels that a system can attain (Sample, 2005). These seven tiers of recovery help identify system fault tolerance level, current risk for data loss, and the target fault tolerance level that a database system can aim for. The recovery levels range from the lowest with no system backups to the highest with complete, immediate and automatic disaster recovery mechanisms in place. Other database software companies such as Oracle and Microsoft have different terminologies for the disaster recovery capabilities of their database software products as compared to what has been classified by IBM, but their disaster recovery approaches are similar to those defined by Sample (2005):

Tier 0: No off-site data, no backup hardware, and no disaster recovery plan

There is no saved information, no documentation, no backup hardware, no contingency plan, and no disaster recovery plan for this solution. The recovery time in this case is unpredictable. In fact, it may not be possible to recover the database system at all. The database system is usually guarded solely by implementing redundant array of independent disks (RAID) and relies on database backups that are kept on site.

Tier 1**:** Data backup with no Hot Site

Under this database disaster recovery plan, database backups will be sent to an off-site facility for safekeeping. The database backups provide the only means to restore the database. Depending on how often backups are made and how they are shipped from one site to another, there might be several days to weeks of data loss. The only good

thing is that the database backups are secure off-site. However, this tier lacks the systems on which to restore the database should the primary system becomes non-operational.

Tier 2**:** Data backup with a Hot Site

In this case, regular database backups are saved on magnetic tapes or CDs and sent to an off-site facility that hosts a secondary infrastructure (also known as a hot site) where the database can be restored using the database backups in the event of a disaster. This solution will still result in the need to recreate several hours to days worth of data, but it is less unpredictable in recovery time.

Tier 3**:** Electronic Vaulting

Tier 3 solutions utilize the hot backup site specified in Tier 2. Additionally, mission-critical data and system backups are vaulted (or copied from the primary site to the backup site) via an automated process. This electronically vaulted data is typically more current than the backups that are shipped using physical media. Electronic vaulting or automated data replication can be implemented by using data mirroring technology from network attached storage (NAS) equipment vendors such as EMC, Network Appliance, Sun, or IBM.

Tier 4**:** Point-in-time copies

Tier 4 solutions are used by businesses that require both greater data currency and faster recovery time than what are provided by database disaster recovery methods from the lower tiers. Rather than relying on backup tapes, CDs, or database backup files being sent electronically from the primary site to the backup site, Tier 4 incorporates more real time solutions. It is much more efficient to recover the database from point-in-time

copies of the database using data changes captured in log files that are replicated from primary site to the backup site than from database backups. Point-in-time copies of database are available using IBM's Batch/Online Database Shadowing and Journaling, Oracle's Data Guard, Microsoft's Warm Standby Server, and similar technologies provided by other database software vendors.

Tier 5**:** Transaction integrity

Tier 5 solutions are used by businesses with a requirement for consistency of data between primary site and backup site. There is little to no data loss for such solutions. This approach is often referred as two-phase commit in which a transaction has to be successfully committed at both primary and backup database sites or the transaction must be totally discarded. This functionality may ensure the system integrity across the sites, but it could severely impact database system performance because of additional workload in the coordination of data integrity check involved between the sites. This functionality is normally a feature of the database software product and usually encoded in the database application.

Tier 6**:** Zero or little data loss

Tier 6 database disaster recovery solutions maintain the highest levels of data currency. They are used by businesses with little or no tolerance for data loss and those who need to restore data to applications rapidly when there is a disruption in database operations. The solutions in this tier would have no dependence on the applications to provide data consistency. This tier includes all of the capabilities described in tier 5 and lower. In addition, it introduces system clustering enabled by operating systems (such as

Sun Clusters) or hardware/software clusters. For database systems running on clusters, when one of the nodes on the clusters fails, there are still other nodes available to carry on the system operations without any disruption. Over the years, cluster has been a proven technology and is offered by hardware and software vendors such as Sun, HP, Dell, Oracle, and Microsoft.

Tier 7**:** Highly automated, business-integrated solution

Tier 7 solutions include all the major components being used for a Tier 6 solution with the additional integration of automation. The solutions in this tier ensure consistency of data above that of which is accomplished by Tier 6 solutions. The database recovery process is fully automated. Restoration of database applications and systems could progress much faster and more reliable than would be possible through manual disaster recovery procedures. These 'turnkey' solutions, however, may not be available out of the box but have to be implemented by integrating many different systems and technologies.

2.5 Research Methods

The research on database system high availability was based on published research papers, case studies, and books on high availability system architecture and technologies. Information on the subject was also gathered from data sheets, white papers, technology tutorials, product specifications, implementation guides, and training materials provided on web sites by database software vendors such as Microsoft, Oracle, and IBM. Relevant information on hardware/software cluster and data replication technologies was obtained from major hardware, software, and storage area network

(SAN) vendors such as Sun, HP, EMC, and Network Appliance. Personal experience on database system design, development, implementation, maintenance, and administration also contributed significantly to this research subject.

Data collected were generated based on search categories and key words related to database fault tolerance and disaster recovery subject. Relevant information were selected and classified into different categories for analysis and comparison that could be used to support the project goal and objective.

2.6 Research Summary

A database system built for high availability demands that the system must have some kind of fault tolerance and disaster recovery capabilities to withstand both planned and unplanned outages and to ensure uninterrupted system operations. The system outages could be caused by natural disasters or due to routine system maintenance tasks such as system backups, or hardware / software upgrades. In today's e-business environment, many companies depend on database systems to provide continuous system availability and transparent disaster recovery ability for business survival. Economically speaking, the cost of a system outage sometimes far outweighs the cost a disaster recovery solution that can be put in place to prevent or minimize the impact of system outages in the first place. The goal of building database system with fault tolerance and disaster recovery capabilities is to use all available resources, establish processes to minimize planned and unplanned system downtime to protect business system operations from unforeseen disasters. A high availability database system usually relies on good database backups, system redundancy, and well-established disaster recovery procedure.

Ultimately, planning for any type fault tolerance and disaster recovery solution for a database system is always subject to balancing system downtime versus cost.

Chapter 3 – Project Methodology

This project utilized waterfall methodology in which the project proceeded from one phase of the Systems Development Life Cycle (SDLC) to the next in a sequential order. The project included the following phases: planning, analysis, design, and implementation phase as depicted in the SDLC graph below:

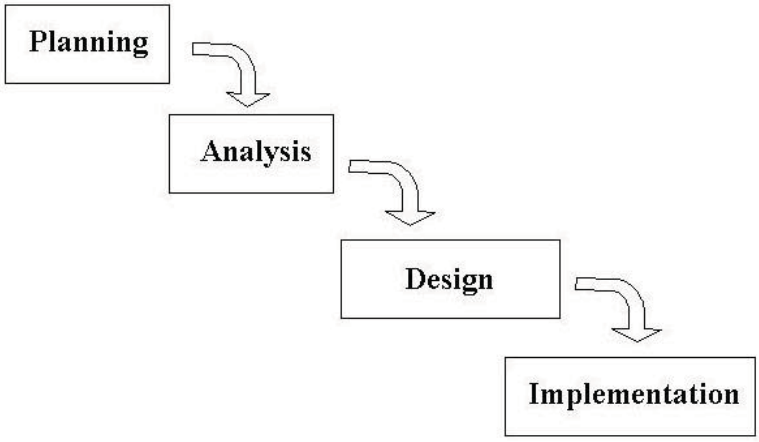

Figure 3: Systems Development Life Cycle (SDLC)

3.1 Planning

The objective of this phase was to determine the business needs and identify possible disasters that could happen to a database system. The outcome of this phase was documentation to analyze capabilities that the database system should have to serve the business needs of the DOJ program office and to identify the risks that might affect database system operations so that preventative measures could be planned, developed, and implemented to eliminate the risks as much as possible. The more preventative measures that could be built into the database system, the lower the chance that the

system operations would be disrupted or damaged such that the database might not be recovered when a disaster strikes. On the other hand, no matter how sophisticated the database system was to be built with all of the preventative measures, there might always be residual risks of system disruption and outage.

3.2 Analysis

In the analysis phase, the risks of the system are identified to define the capabilities of the system to handle the perceived risks to the system. The objective of this phase was to document the risks, specify system requirements, and develop system disaster recovery plan. Appendix A provides the set of questions which was used to gather the system requirements for the new database system architecture.

The system requirements gathered and analyzed in this phase were used to determine disaster recovery level required for DOJ application systems. The outcome of the analysis phase was to provide criteria to select the technology for disaster recovery solutions for DOJ application systems.

3.3 Design

The objective of this phase was to provide a system architecture design that could be implemented to provide disaster recovery and fault tolerance capabilities for DOJ application systems as specified in the requirements analysis phase. The design of a fault tolerance database system is partially dependent on the hardware and software solutions. The solutions generally are similar and comparable with the current available technologies offered by major database software vendors. Open source database software products can provide viable solutions; however, they may not have all of the desired

features or options available as other well-known commercial database software products.

The design of database system architecture is certainly driven by the database software products and solutions selected. However, the design may not be dependent on the hardware products. The reason is that commercial database software products can run on many different hardware platforms. Furthermore, hardware products normally provide similar functionalities to support high availability database system architecture. A database system architecture designed independent of database software product is provided in Chapter 4.

3.4 Implementation

The implementation phase consisted of tasks performed to build the designed database system architecture. The tasks were developed using instructions and guidelines from technical documents provided by database software vendors. It is a common practice to build a development environment that is similar to the production environment to develop and test the implementation procedure and to ensure that the system works as expected before performing the implementation procedure in the production environment.

The database system architecture design was divided into two sections and implemented in two independent stages. The first stage was to implement the production database environment. The second stage was to implement the standby database environment as well as to integrate the standby database into the system architecture. Each implementation stage was further broken up into specific tasks to be performed. The implementation plan followed the approach to build and test each system components independently, then integrate the components together as the project was

progressed. A detailed implementation plan which lays out the tasks for project implementation is provided in Chapter 4.

Chapter 4 – Database System Architecture for Fault Tolerance and Disaster Recovery

4.1 Database System Architecture Design

The design of database system architecture is certainly driven by the database software products and solutions selected. This section will look at the solutions for database system high availability and disaster recovery offered by two major database software vendors Microsoft and Oracle for comparison.

Microsoft SQL Server database offers the following database fault-tolerance and disaster recovery capabilities (Microsoft, 2007):

1. Failover Clustering
2. Database mirroring
3. Peer-to-peer transactional replication
4. Maintenance of a warm standby database
    - By log shipping
    - By transactional replication
5. Backup and restore
6. Disk redundancy of data by using redundant array of independent disks (RAID)

The following table provides a brief description of disaster recovery options for Microsoft SQL server database along with the advantages and disadvantages of each option:

Table 4: Microsoft SQL Server Disaster Recovery Options

Disaster Recovery Options	**Functionality**	**Advantage**	**Disadvantage**
1. Failover Clustering	Cluster of database servers to provide database failover automatically if hardware failure or a software failure occurs	- High server availability with almost no downtime - Automatically occurs if the primary server fails	- Install and maintain two servers is two times the costs of maintaining a single server - Servers should be in the same location
2. Database mirroring	A mirror database set up at a backup site	- Increase data protection - Increase system availability - Improve the availability of production database during upgrades	- Mirror database should be identical to the principal database - The transfer of information from one server to another over a network must be secure
3. Peer-to-peer transactional replication	Application can read or modify the data in any database that participates in replication	- Improve read performance - Aggregate update performance, insert performance, and delete performance for the topology resembles the performance of a single node because all changes are propagated to all nodes	- All participating databases must contain identical schemas and data - Peer-to-peer transactional replication does not provide conflict detection or conflict resolution

4. Warm standby database server	Maintain warm standby server by using either of the following methods: • Log shipping • Transaction replication	• Log shipping - Can recover all database activities - Restore database faster • Transaction replication - Can read data on standby database - Changes are applied with less latency	• Log shipping - Database is unusable during recovery process - Lack of granularity - No automatic failover • Transaction replication - Switching servers erases replication configurations - If disaster occurs, must manually switch servers
5. Backup and restore	Use backup copy to re-create the database or to restore the database	- Can back the database up to removable media - Do not depend on network as failover clustering or log shipping	- If failure occurs, may lose your most recent data - If disaster occurs, must manually restore database
6. Disk redundancy of data by using redundant array of independent disks (RAID)	RAID stores redundant data on multiple disks to provide greater reliability and less downtime for servers	Do not lose data if any one disk fails	- It may take a long time to recover the data - If multiple disks fail, may not be able to recover valuable data

The following are Oracle software products and features that deliver similar database disaster recovery options:

1. Real Application Clusters (RAC): Database failover clustering
2. Data Guard: Database mirroring, Physical and Logical Standby Database
3. Oracle Streams: Peer-to-peer transactional replication
4. Recovery Manager (RMAN): Backup and restore
5. Disk redundancy of data by using Automatic Storage Management (ASM) and redundant array of independent disks (RAID)

The following figure depicts one of the database system architecture designs for fault tolerance and disaster recovery that can be implemented independent of the database software products.

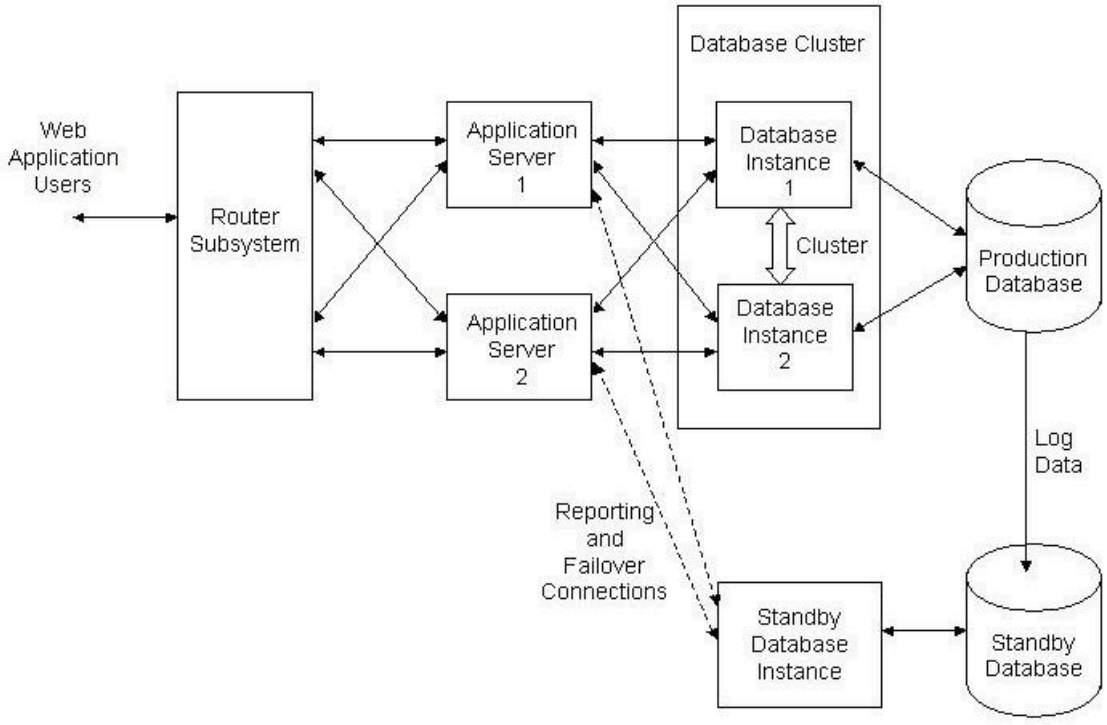

Figure 4: High Availability Database System Architecture

In this database system architecture, redundant application servers and database servers are used to provide non-interrupted service should one of the servers be damaged and cannot immediately be repaired. The database servers are also clustered to take advantage of the database cluster technology allowing high availability and scalability to the database system. A standby database is also utilized to provide more protection to the database system. The standby database is normally located at a different site as a safeguard in case the production site is totally disrupted. The standby database may possibly be used to generate reports for applications to take some of the workload off the production database. Other options can be derived from this design as a subset of it by taking away those components that are deemed not required or can't be afforded. On the other hand, more nodes can be added to the clustered servers, as well as additional standby database and application servers can be set up at multiple backup sites as required to handle the heavy workload and to attain maximum availability for the system.

4.2 Oracle Database System Architecture Design

This section provides a system architecture design specifically for Oracle database technology using the guidelines from the Oracle Database High Availability Best Practices document on how to implement high availability database system architecture for Oracle database.

The table below outlines Oracle products and features that may be implemented within database system architecture to handle planned or unplanned database system outages (Oracle, July 2006):

Table 5: Oracle Database High Availability Architecture

Outage Type	Database Capabilities and Features
Unplanned	
Computer failures	• Oracle Database 10g with RAC • Oracle Database 10g with Data Guard • Oracle Database 10g with Streams • Fast-Start Fault Recovery
Storage failures	• Automatic Storage Management • Recovery Manager (RMAN) • Flash Recovery Area
Human errors	• Oracle Security Features • Oracle Flashback Technology • LogMiner
Data corruption	• Block Checking • Block Checksumming • Hardware Assisted Resilient Data (HARD) Initiative • Oracle Database 10g with Data Guard • Oracle Database 10g with Streams • Recovery Manager • Flash Recovery Area
Site failures	• Oracle Database 10g with Data Guard • Oracle Database 10g with Streams • Recovery Manager (RMAN)

Outage Type	Database Capabilities and Features
Planned	
Data changes	• Online Reorganization and Redefinition • Oracle Database 10g with Data Guard • Oracle Database 10g with Streams
System changes	• Automatic Storage Management • Dynamic Resource Provisioning • Rolling patch updates and system upgrades using Oracle Database 10g with RAC • Rolling release upgrades and system upgrades using Oracle Database 10g with Data Guard or Oracle Database 10g with Streams • Platform Migrations and Database Upgrades with Transportable • Tablespace

Based on Oracle database functionalities and features provided in the above table, Figure 5 shows an implementation approach for a high availability database system architecture consisting of clustered database with a standby database for Oracle database software products.

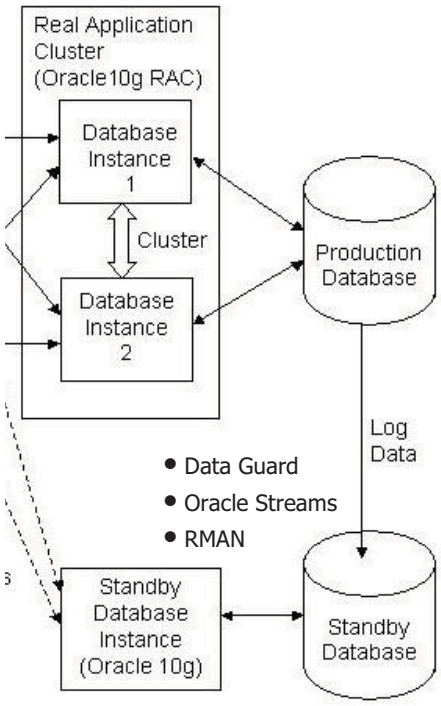

Figure 5: High Availability Oracle Database System Architecture

In Figure 5 above, Oracle 10g Real Application Clusters (RAC) is used to provide system redundancy for the database servers. The database servers are clustered. There is one database instance running on each of the clustered servers. The database instances can be set up in an active-active configuration for load balancing in which the database instances alternatively receive and process service requests from the web servers. Another option is to have the database instances set up in an active-passive configuration for system fail-over in which only one database instance is up (also known as the primary instance) and the other instance (secondary instance) is down but would be brought up automatically in the event that the primary instance is not functioning. DOJ program office would like to implement active-active configuration for the clustered database

servers for load balancing. In either case, both database instances share one set of database files normally located in a network storage array (NAS). The production database system is disrupted only when both database instances are not functioning. A standby database is used to receive the log files from the production database and serves as a backup database in the event that the production database system is totally in-operational. The database log files are transferred from the production database to the standby database using Oracle Streams or Data Guard feature. The standby database would be activated using Oracle Recovery Manger (RMAN) feature. When the standby database is activated, the web servers will be re-configured to connect to the standby database so that the system can operate normally while the production database environment is repaired. The activation of standby database is usually performed manually, but it can be automated if the criteria for the activation are well defined and can be programmed. For better system protection, the standby database should be located in a different geographical area from the production database site and safeguards should be put in place to reduce the chance that both production database site and standby database site getting hit by the same disaster be it natural or man made. This database system architecture ensures high availability with zero or little data loss. The only scenario that this database system architecture is out of service is when the production database servers and the standby database server are all in-operational.

Figure 6 provides technical details on how primary Oracle database communicate with the standby database in the database system architecture (Oracle, September 2006).

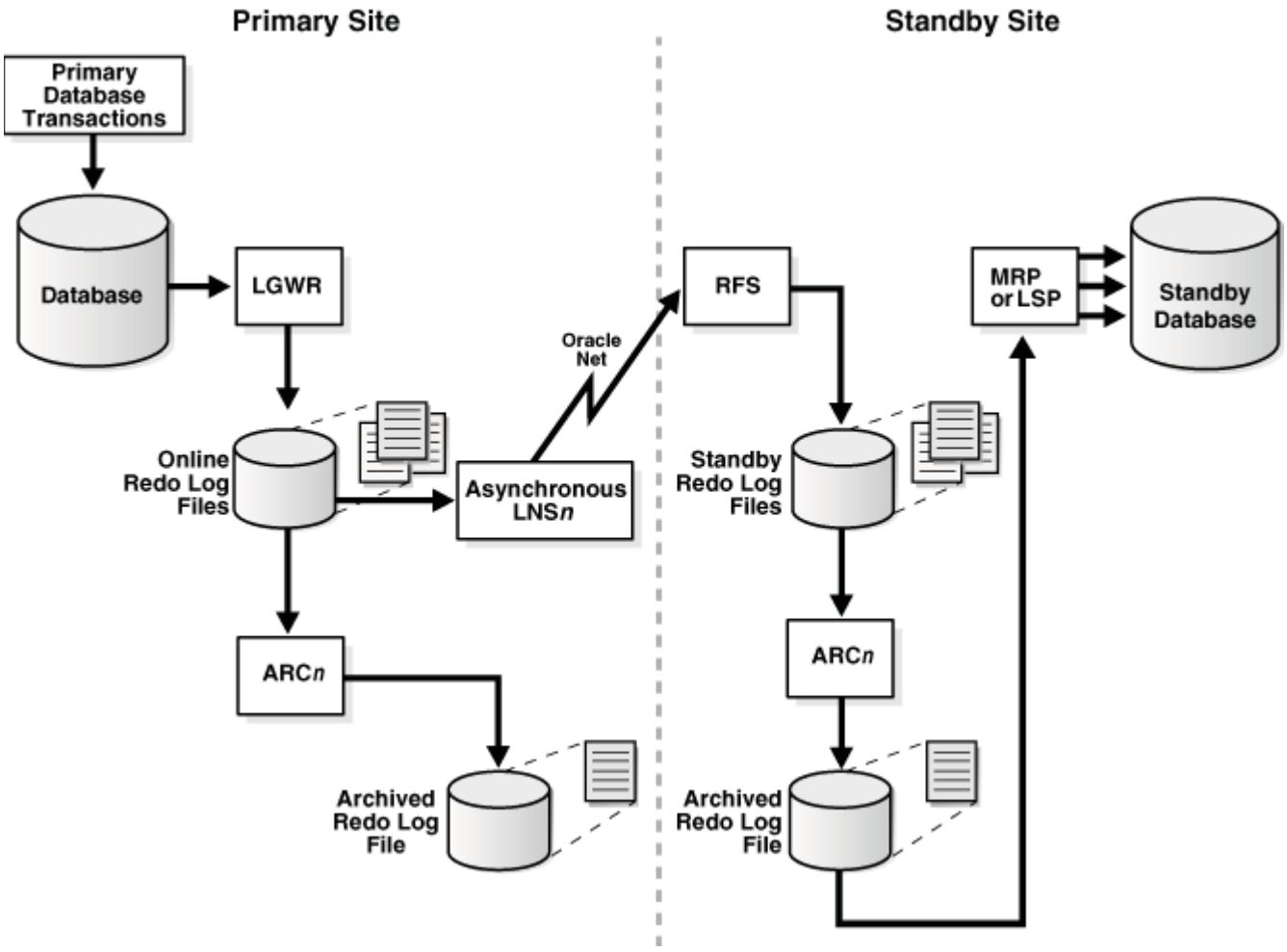

Figure 6: Oracle Standby Database Implementation

Legend:

LGWR – Log Writer process writes redo log buffer from memory to online redo log file
ARCn – Archive processes archives redo log files
LNSn – LGWR Network Server processes
RFS – Remote File Server process receives archived redo logs from primary database
MRP – Managed Recovery Process applies information from archived redo logs
LSP – Logical Standby Process applies completed SQL transactions from archived redo logs

4.3 Implementation Plan

The implementation plan composes two stages. The first stage is to implement the clustered database servers. The second stage is to implement the standby database and integrate it into the database system architecture. The system architecture design allows the two stages to be built independently. Due to schedule constraints, DOJ program office decided to build and operate the system with just the database cluster for initial operational capability. The database cluster by itself can provide some degree of fault tolerance to the system. When the database cluster is operational in production, the program office will build and implement the standby database at another location, then integrate it into the system architecture. Since the production clustered database servers and the standby database server are located at different sites, it would take some time to work out the network communications for data transfer between the servers and enterprise system security issues at both sites.

This section only describes the implementation plan for the first stage which is to implement the clustered database servers. The implementation plan provides an outline of database upgrade and migration tasks for each of the three DOJ application systems. The tasks involve upgrading database software from Oracle 9i to Oracle 10g and migrating existing database to a new infrastructure while ensuring that the current applications work with the new database system. A team of one database administrator, one system administrator, and two application developers is required to work on system migration and upgrade for the entire system architecture. Migration of application systems will be done at the same time but sequentially in the order determined by system priority.

The implementation plan for each application system consists of three main tasks:

1. To set up an environment to perform tests and migration of Oracle production database from version 9.2 (Oracle 9i) to version 10.2 (Oracle 10g) without modifying the existing applications.
2. To build infrastructure of redundant web servers, application servers, and configure clustered database servers to maximize the availability and survivability of system operations.
3. To migrate applications and production database onto the new infrastructure.

It is estimated that the tasks will be completed in 18 weeks for all three application systems.

4.3.1 Overview

An Oracle database consists of two main components: database instance (which includes background processes running in the system memory) and database files (which reside on physical disks.) When the database runs on a single sever, hardware problems such as processor failure, memory failure, or disk failure can stall the database operation. In the Oracle Real Application Cluster (RAC) configuration, database instances are set up across multiple servers or processors. These server processors are clustered to allow robust process management and scalability. The cluster can be expanded by adding additional servers to it to get more processing power. Database files of a RAC database normally reside on Redundant Arrays of Inexpensive Disks (RAID) or on Network Attached Storage (NAS). The RAID and NAS provide files protection and storage expansion capability.

The current application system architecture consists of one application server and one database server. This configuration does not provide system fault tolerance. System operation can be disrupted when either application server or database server fails.

The new system architecture consists of redundant application servers, Oracle Real Application Cluster database running on Sun cluster, and database files residing on network attached storage. The new architecture is designed to be fault tolerant and scalable. It also provides flexibilities including:

- Application servers can be added to the architecture to handle additional transactions.
- Servers can be added to expand the cluster to provide more processing power to the clustered database.
- Disks can be added to NAS to provide more storage area for database files.

4.3.2 Scope

The tasks in this implementation plan involve database and application server software installation, configuration, and testing. The scope and objectives of these tasks are as follows:

- To set up a test environment to perform database migration and upgrade testing.
- To configure and test new system architecture which includes redundant application servers, Oracle Real Application Cluster database on Sun cluster, and standby database.
- To upgrade the current production database on a single server environment.

- To migrate the upgraded production database from single server environment to the new system architecture.

These tasks will be performed under the assumption that the hardware, operating system, and network architecture are already set up and functioning.

4.3.3 Current System Architecture

The following diagram depicts the current system architecture for the application system to be upgraded:

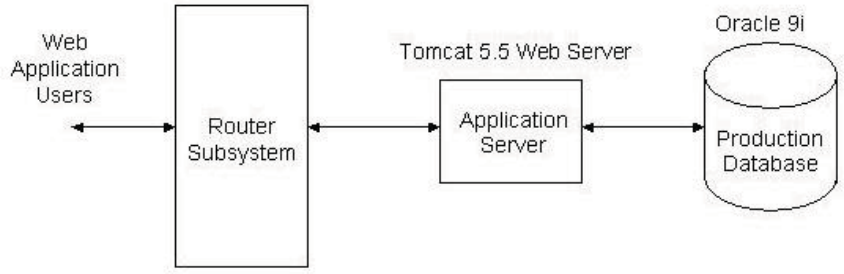

Figure 7: Current System Architecture

- Hardware

The hardware of the current system architecture consists of a database server and an application server. Both servers are Sun450 servers. Each server has 1GB RAM and 4x9GB hard disks. There are no redundancy or dedicated backup servers in the current system architecture.

- Software

The current production database is running on Oracle 9i Enterprise Edition (version 9.2). The applications are written in Java, currently running on Tomcat 4.0 application server. The users access the applications through web browsers on their

computers.

4.3.4 Proposed System Architecture

The proposed system architecture to be implemented for the first milestone is depicted in Figure 8 below. It is a subset of the database system architecture for fault tolerance and disaster recovery.

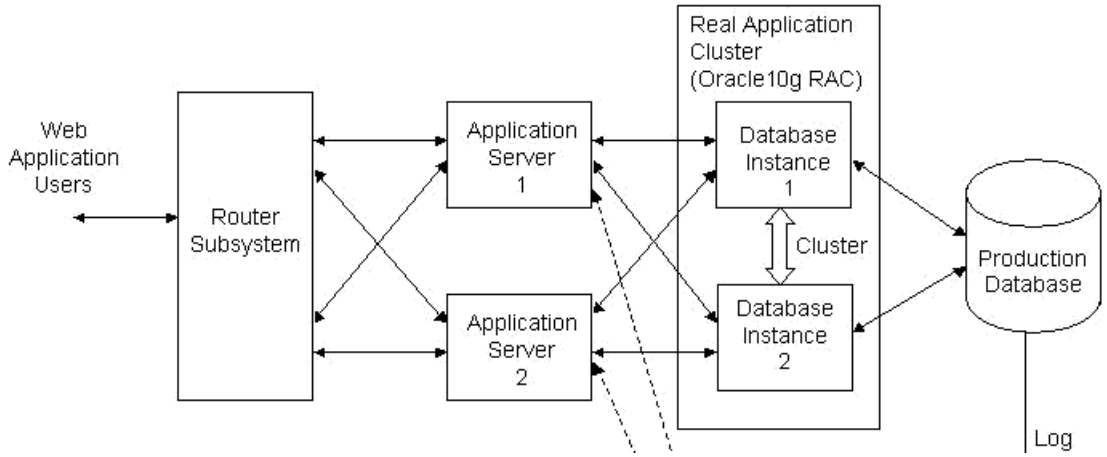

Figure 8: Proposed System Architecture

- Hardware

The database for the proposed system architecture depicted above runs on a two-server cluster. The servers are Sun V880 servers. Each server has 2GB RAM and 4x9GB hard disks. The application servers consist of two redundant servers to provide load balancing and fail protection for the system. The servers are Sun V880 servers. Each server has 2GB RAM and 4x9GB hard disks. Network Attached Storage (NAS) provides disk space for RAC databases. The total disk storage capacity of NAS is 2 tetrabytes. NAS is provided by Network Appliance (NetApp) servers and configured by NetApp technicians under a product support contract.

- Software

The database for the proposed system architecture runs on Oracle 10g Enterprise Edition (version 10.2.) on Solaris 10 operating system. The current Java applications will be migrated to Oracle 10g Application Server on Solaris 10 virtual servers.

4.3.5 Database Migration and Upgrade Test Plan

- Software Installation

The following are the tasks to be performed:

- Install Oracle 10i Enterprise Edition database software on the new development and test database server
- Install Oracle 10g Enterprise Edition database software on the new production database server

- Setup and Configuration

The following are the tasks to be performed:

- Create and configure Oracle 10g database on the development and test database server
- Create and configure Oracle 10g database on the production database server
- Configure the test applications to work with Oracle 10g test database and Oracle 10g test database for parallel testing
- Develop procedures to migrate Oracle 9i database to Oracle 10g database

- Testing

The following are the tasks to be performed:

- Database upgrade from Oracle 9i to Oracle 10g procedures
- Parallel testing on test applications on Oracle 9i and Oracle 10g databases

4.3.6 System Architecture Test Plan

- Software Installation

The following are the tasks to be performed:

- Install and configure Solaris 10 and Oracle 10g Real Application Server (Oracle 10g RAC) on the clustered Sun V880 servers
- Install and configure Solaris 10 virtual servers and Oracle 10g Application Server on the redundant Sun V880 application servers
- Install Solaris 10 and Oracle 10g Enterprise Edition database software on the standby database server

- Setup and Configuration

The following are the tasks to be performed:

- Create database instances on Oracle 10g RAC servers
- Configure the database instances to work with database files on the storage area network
- Configure the Oracle 10g Application Servers to work with the database instances on the Oracle 10g RAC servers
- Create and configure Oracle 10g standby database on the standby database server

- Testing

The following are the tasks to be performed:

- Oracle 10g RAC architecture testing
- Standby database testing
- System architecture testing

- Database fail-over testing
- Database backup and recovery testing

4.3.7 System Cut-over

System cut-over will be carried out in two phases:

- Phase One: Upgrade the database from Oracle 9i to Oracle 10g on single server configuration
- Phase Two: Migrate Oracle 10g from single server configuration to clustered servers configuration

4.3.8 Timeline

The following graph provides a timeline of the implementation tasks to be performed:

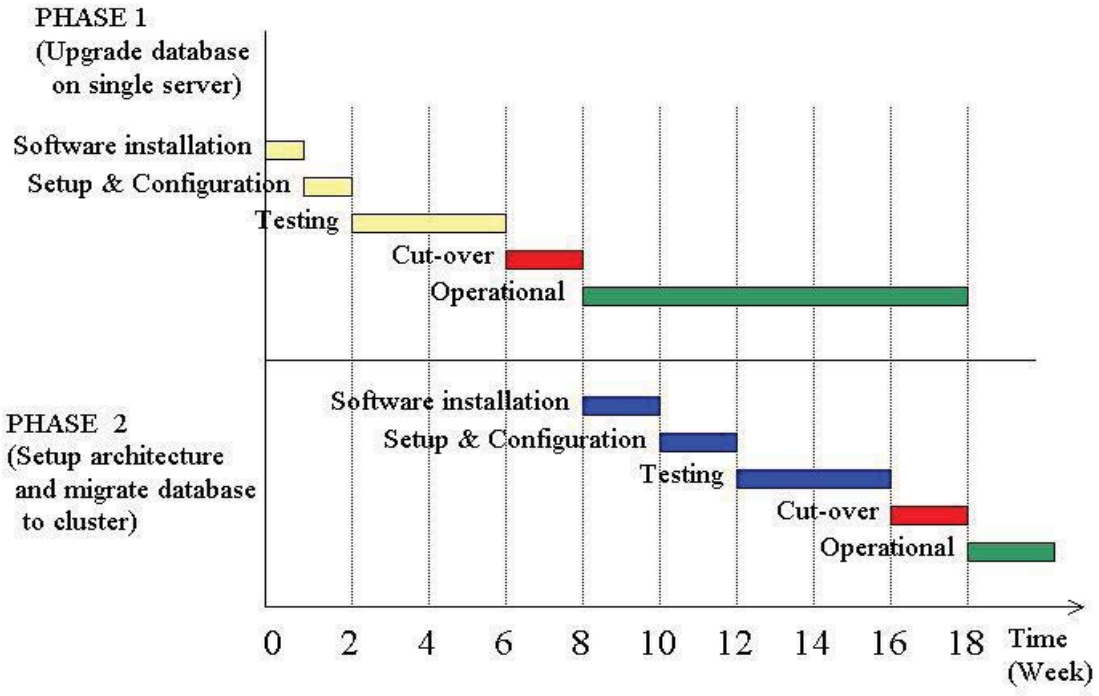

Figure 9: System Implementation Timeline

4.4 Implementation Procedures

This section provides procedures to implement Oracle 10g Real Application Clusters (RAC) and to perform database upgrade from Oracle 9i to Oracle 10g RAC on Sun Solaris servers as part of the implementation plan for the database system architecture described in section 4.3.

4.4.1 Oracle 10g RAC Software Installation

The installation of Oracle 10g RAC software includes Oracle 10g Clusterware and Oracle 10g Database software.

a. Install Oracle 10g Clusterware

Follow the instructions in the Oracle Database Clusterware and Oracle Real Application Clusters Installation Guide 10g Release2 (10.2 for Solaris Operating System document to perform pre-installation tasks to prepare for the installation of Oracle 10g clusterware. When the pre-installation tasks are all done, follow these procedures to install Oracle Database 10g clusterware:

1. Verify that the kernel parameters in the /etc/system file are set to values greater than or equal to the recommended value:

Parameter	**Recommended Value**
noexec_user_stack	1
semsys:seminfo_semmni	100
semsys:seminfo_semmns	1024
semsys:seminfo_semmsl	256
semsys:seminfo_semvmx	32767
shmsys:shminfo_shmmax	4294967295
shmsys:shminfo_shmmin	1

Parameter	Recommended Value
shmsys:shminfo_shmmni	100
shmsys:shminfo_shmseg	10

2. Run Cluster Verification Utility (CVU) to verify whether the servers are ready for Oracle clusterware installation by executing the following command on one of the servers to be clustered:

   [oracle@tis06db cluvfy]$ <ORACLE_HOME>/runcluvfy.sh comp nodereach -n node1,node2 -verbose

3. Run Oracle Universal Installer to install Oracle Clusterware

   ```
   $ <Oracle10g_clusterware_dir>/runInstaller
   ```

4. Welcome screen (Click on Next)
5. Specify Inventory Directory and Credentials screen

   - Enter inventory directory path and OS group name (Click on Next)

6. Specify Home Details screen

   - Enter name and path for cluster home (Click on Next)

7. Product-Specific Prerequisite Checks screen (Click on Next)
8. Specify Cluster Configuration screen

   - Enter Cluster Name

   - Click “Add” and provide public, private, and virtual node names for cluster nodes (Click on Next)

9. Specify Network Interface Usage screen

   - Select Private and Public interfaces (Click on Next)

10. Specify Voting Disk Location screen

    - Check box for Normal Redundancy

    - Enter directory path for Voting Disks (Click on Next)

11. Summary screen (Click on Install)

b. Install Oracle Database 10g Release 2 Software

After installing Oracle Clusterware, follow the instructions in the Oracle Database Clusterware and Oracle Real Application Clusters Installation Guide 10g Release2 (10.2 for Solaris Operating System document to perform pre-installation tasks to prepare for the installation of Oracle Database 10g software. When the pre-installation tasks are all done, follow these procedures to install Oracle Database 10g software:

1. <Oracle10g_sw>/runInstaller
2. Select Installation Type screen appears.

    - Installation Type: Enterprise Edition (Click on Next)

3. Specify Home Details screen

    - Enter Oracle home name and path (Click on Next)

4. Specify Hardware Cluster Installation Mode screen

    - Check box for Cluster Installation (Click on Next)

5. Product-Specific Prerequisite Checks screen (Click on Next)
6. Select Configuration Option screen

    - Check the button for Install database Software only

(Click on Next)

7. Summary screen (Click on Install)

4.4.2 Oracle Database Upgrade from 9i to 10g

This procedure describes the steps to create an Oracle 10g database using the Database Configuration Assistance (dbca) utility, export data from Oracle 9i database, and import the data into Oracle 10g database.

a. Create Oracle 10g database

1. Run Database Configuration Assistance (dbca) utility

   $ dbca

2. Welcome screen appears (Click on Next)
3. DBCA Operations screen
   - Select Create a Database (Click on Next)
4. DBCA Template screen
   - Select Custom Database (Click on Next)
5. DBCA Database Identification screen
   - Enter Global Database Name and SID (Click on Next)
6. DBCA Management Options screen
   - Select Configure the Database with Enterprise Manager

   (Click on Next)
7. DBCA Database Credentials screen
   - Enter password: [enter password] (Click on Next)
8. DBCA Storage Options screen
   - Select File System (Click on Next)

9. DBCA Database File Locations screen

   - Select Use Database File Locations from Template

   (Click on Next)

10. DBCA Storage Options screen

   - Select Specify Flash Recovery Area and enter a directory

   location for it (Click on Next)

11. DBCA Database Content screen

   - Do not check Sample Schemas (Click on Next)

12. DBCA Initialization Parameters screen

   - Memory tab:

      . Select Typical

      30 percent of physical memory

   - Connection Mode tab:

      . Select shared server mode and share server: 10

      (Click on Next)

13. DBCA Database Storage screen

   - Enter the directory and file names for database files

   (Click on Next)

14. DBCA Database Creation Options screen

   - Select Create Database

   - Select Save as a Database Template

   - Select Generate Database Creation Scripts

   (Click on Finish)

b. Create Tablespaces

Create tablespaces that exist in production database for the Oracle 10g database.

c. Database Export and Import

Perform database export from the production database (Oracle 9i database) and import the data into the Oracle 10g database.

Chapter 5 – Project History

5.1 How the project began

This project was based on a system development task order searching for solutions to implement high availability database system architecture to migrate three existing database application systems from single server architectures to fail-safe consolidated database system architecture for the DOJ program office. The project mainly focused on the available technologies provided by major database software vendors that can be recommended to the DOJ program office funding the project.

Research for the project took about three months from March to May 2007. Project follow-on work as outlined in the project plan started in June 2007 and was completed in December 2007. Each phase was successfully executed according to plan and met the scheduled timeline.

5.2 Changes to the project plan

The original goal of the project was to present different technologies and system architectures provided by the major database vendors such as Oracle, IBM, and Microsoft for comparisons before recommending for a solution. However, there were so much research materials gathered on the subject from these database vendors and other hardware and software vendors that they became overwhelming and posed some risks of getting the project running out of scope and out of schedule. Therefore, rather than going in depth with all available database products, only Oracle database technology was used to develop activities involved in the design, development, and implementation of high availability database system architecture. Furthermore, the DOJ program office heavily

leaned toward Oracle database software products to ease the learning curve for its technical staff and to speed up the implementation timeline.

5.3 What went right and what went wrong

One of the utmost concerns for this project was to narrow down the scope of the project and to get the problem solved within a reasonable amount of time that the DOJ program office had set out. Throughout the project, the scope of the project was kept in focus by searching for workable solution and narrowing the research to find relevant materials supporting the project goal. A considerable amount of time spent in the research and a lot of research materials were gathered and analyzed. Nevertheless, the research effort deemed to be very worthwhile. Considerable research material had not been included so that it would not distract from the main objectives or to run the project out of scope. Decisions had been made quickly on defining system requirements and to select a workable solution to complete the project on schedule.

5.4 Project variables and their impact

This project aimed to build a small scale database system architecture to keep the scope of project in focus while still maintain the complexity of designing, developing, and implementing a high-availability database system architecture. The simple system design presented can be easily scaled up to include additional servers or operational sites as required. The design can also be scaled down to just include must-have functionalities and to forgo those options that may not be needed or could not be afforded - to keep costs within budget if funding was one of the deciding factors. The personnel needed to carry out the project plan would depend on the scale of the project such as the number of servers and sites to be implemented to handle specific needs of the organization.

5.5 Results summary

The project addressed major concerns and offered realistic solution for building high-availability database system architecture. It also provided detailed analysis, system design, development, and implementation that could be readily applied and tailored for practical business application systems especially for those systems utilizing Oracle database technology. Customized system design and implementation can certainly be derived from the analysis and study presented in the project paper. The project achieved the main objectives and delivered effective system architecture design for high availability database system architecture for the DOJ program office.

Chapter 6 – Lesson Learned

6.1 What was learned from the project experience

There were many things that had been learned throughout the project such as project management, time management, organization, and decision making process. Other important skills that had been also acquired or enhanced include research, documentation, and presentation. The project also offered an opportunity to study cutting edge and advanced technologies in database related hardware and software from various database and systems vendors.

6.2 Did project meet initial project expectations

The project was set out to present different database technologies for fault tolerance and disaster recovery. Though technologies offered by database vendors may be different in implementation, they all follow the same common disaster recovery strategy. A great percentage of time involved was devoted in studying a particular database software product that the DOJ program office had selected. An implementation plan was successfully developed and executed. This project may be considered a case study for someone who may be looking for such a solution. The material provided in the project paper aim to make the goal of the project succinct, practical, valuable, and worthwhile to study.

6.3 Next stage of project if it continued

The next stage is to develop the implementation plan for the second stage of the project which is to implement the standby database and integrate it into the system architecture. Overall, this project can certainly be applied to other business case scenarios that have similar objectives. It may require a considerable investment of

financial and human resources to implement the proposed database system architecture in a larger scale. A team consisting of ten or more system analysts, system engineers, database administrators, and application developers may be needed to implement larger scale database system architecture. There is also a need for system maintenance and upgrade after implementation to keep the system up to date with the advance of technologies.

6.4 Conclusions/recommendations

Developing reliable database system architecture requires great effort to analyze the risks that can affect the operations of the system so that proper measures can be planned and implemented into the system to mitigate the risks of system disruption when a disaster occurs. The system capabilities must be clearly defined in the system requirements analysis. The analysis phase has to be done throughout with all of the perceived system risks identified so that preventive measures and procedures can be developed to minimize the risks. When the requirements in the analysis phase are changed, sometimes it is hard to propagate the change down the road once the analysis phase has been finalized. A particular instance is that it would be difficult to change the system design after the equipment are already purchased for the scale and functions specified in the analysis phase. One of the approaches that can be used to implement this project is to divide the project into modules so that different components can be added to the system and each module can be modified independently without affecting the entire system architecture. The system architecture should be designed to be scaled up for additional capabilities and functionalities as much as possible.

References

Boukerche, A. et al. (2007). *Towards highly available and scalable high performance clusters*. Journal of Computer and System Sciences. 73(8), 1240-1251 Retrieved March 5, 2008 from http://portal.acm.org/citation.cfm?id=1296454

Brooks, Charlotte. (2002). Disaster Recovery Strategies with Tivoli™ Storage Management. IBM Redbooks.

Bruni, Paolo et al. (2004). Disaster Recovery with DB2™ UDB for z/OS. IBM Redbooks.

Burleson, Donald K. (2005). *Costs and Benefits for High Availability Oracle9i*. Retrieved April 5, 2007 from http://www.dbazine.com/oracle/or-articles/burleson4/view?searchterm=reliability

Marcus, Evan and Hal Stern. (2003). *Blueprints for High Availability, Second Edition*. Indianapolis, IN: John Wiley & Sons

Microsoft. (2007). *Description of Disaster Recovery Options for Microsoft SQL Server ™*. Retrieved April 17, 2007 from http://support.microsoft.com/kb/822400#

MySQL.com. (2007). MySQL™ Cluster. Retrieved May 6, 2007 from http://www.mysql.com/products/database/cluster/

Oracle. (July, 2006). *Oracle™ Database High Availability Best Practices*. Retrieved April 20, 2007 from Oracle Technology Network website: http://download.oracle.com/docs/cd/B19306_01/server.102/b25159.pdf

Oracle. (September, 2006). *Oracle® Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide 10g Release 2 (10.2) for Solaris Operating System*. Retrieved August 23, 2007 from http://download.oracle.com/docs/cd/B19306_01/install.102/b14205.pdf

Russel, Charlie, Sharon Crawford, and Jason Gerend. (2007). Microsoft Windows Server™ 2003 Administrator's Companion, Second Edition. Microsoft Press.

Sample, Dennis H. (2005). *Prepare for Business Survival: Continuous Availability and Disaster Recovery for z/OS, z/VM and Linux on zSeries*. CCR2. Retrieved April 2, 2007 from http://www-306.ibm.com/software/tivoli/features/ccr2/ccr2-2005-02/feature-continuous-availability.html

Stringfellow, Stan. (2000). *Disaster Recovery Requirements Analysis*. Retrieved April 12, 2007 from Sun BluePrints Online website: http://www.sun.com/blueprints/0700/drra.pdf

Sun Microsystems. (2007). Sun™ Cluster Data Sheets. Retrieved May 3, 2007 from http://wwws.sun.com/software/cluster/ds/index.html

Webb, Warren. (2008). Always On. Embedding High Availability. *EDN*, 53(10), 33-38

Wold, Geogrey H., and Robert F. Shriver. (1997). *Risk Analysis Techniques*. Retrieved April 12, 2007 from http://www.drj.com/new2dr/w3_030.htm

Appendix A

System Requirements Questionnaire

The following are questions that were used to assist in gathering information and provide specifications for database disaster recovery requirements analysis for the DOJ applications.

1. What is the disaster recovery classification required for these applications?
2. How much data can the DOJ program office afford to lose when a disaster happens?
3. How much of system performance degradation is acceptable during a disaster?
4. How often would the program office test and validate the disaster recovery functionalities of the system architecture? (once a year, twice a year, or at other predetermined intervals)
5. What criteria determine the disaster recovery to be put in place for these applications?
6. Do these applications send data to or receive data from other applications? How often is the data exchange? How long each data exchange last?
7. Do these applications interact with other database systems using different database software such as SQL Server, Sybase, Informix, DB2, MySQL, etc.?
8. How big are the databases now? How much will they grow in the next six months, one year, two years, and five years?
9. Are databases updated via online application transactions, batch processes, data feeds, etc.?

10. If data loss occurs after a disaster, is there a way to re-enter the data into database via online transaction processing (OLTP), batch processes, data feeds, or other methods?
11. If network bandwidth must be allocated to support a standby database, what is the average data transfer rate in MB per minute to support the archive logs transfer?
12. What file systems are required by the applications (include file systems for software products, application binaries, external feeds, and so forth) that should be included in the disaster recovery procedures?
13. How do users access the production systems? How should users access the systems in a disaster scenario?
14. Is there a need to build a separate and identical architecture for development and test environments?
15. Are development machines going to be used as alternate servers for disaster recovery?
16. Will the development environment need to be up and running during a disaster?