MARSHALL PLAN PROJECT

Enhancement of Active/Active Replication for a Digital Rights Management Product Activation Server

prepared for the
Austrian Marshall Plan Foundation

submitted by:
Armin Thurner

Fachhochschule Salzburg University of Applied Sciences

Salzburg, March 2011

Details

First Name, Surname:	Armin Thurner
University:	Salzburg University of Applied Sciences
Degree Program:	Information Technology & Systems Management
Title of Thesis:	Enhancement of Active/Active Replication for a Digital Rights Management Product Activation Server

Abstract

Since the very first days of software development, companies have to face the fact that they loose a lot of money due to software piracy. With the development of SecuROM™, Sony DADC Austria AG soon became one of the leading companies helping software engineers to protect their intellectual property against illegal and harmful usage.
Due to our fast growing Internet society, companies started to distribute their software products not only on optical media - Sony DADCs core business - but also via digital content downloads. For this reason, Sony DADC started to develop a Digital Rights Management System for digital distribution channels, compatible with the existing solution for optical media.
SecuROM™Product Activation (PA) has soon become one of the most widely used online activation systems and is growing ever since it has been established. Processing thousands of activation requests daily, a powerful database system needs to work in the background. This database system is permanently addressed with different requests, either by end users, customers or administrators. Therefore a highly available infrastructure at a very low error rate is a key component to meet the needs of the parties who are dependent on the systems smooth operation.
To ensure a permanent and stable environment, several servers are set up as a distributed database system world wide. But not only availability is important, another demand for hassle-free operation is data consistency. Given these requirements it is getting more and more important to operate a replicated system which is actively synchronized in a narrow time frame, thus consistent and reliable. There are various rudiments to approach this challenge. In Sony DADCs particular case, after a fundamental analysis of needs, a commitment for the implementation of an Active/Active Replication system has been made.
Due to permanent development of new features for the actual activation system, the replication implementation has suffered and major update as well as performance improvements have to be enforced.

Table of Contents

List of Figures

List of Tables

1 Introduction

Sony DADC is a leading provider of copy control technologies for software applications and video content. A core component of SecuROM^{TM}, a Digital Rights Management (DRM) solution for Software applications, comprises the online activation via a PA Server.

As a matter of fact, it is not possible to run such a system with only one server, therefore several data centers are located around the world and maintained by Sony DADCs headquarter in Austria. Each of these data centers exists of n[1] application servers and *one* shared database server, running an Oracle 10g database.

To ensure a stable and highly available environment, an Active/Active Replication system has been planned. Even though Oracle offers a native replication implementation, it has not been applicable for Sony DADCs purpose.

Security is one of the main reasons why an independent solution was implemented. Oracles solution requires direct database connections which is incompatible with internal as well as customers Information Technology (IT) policies. As another reason, heterogeneous database and scheme versions need to be supported.

In the course of a diploma thesis, an Active/Active Replication system has been implemented; but due to parallel database and application updates, the replication implementation has soon become inapplicable and needs to be adapted.

[1] At least one but preferable two or more

1.1 Motivation

The rapidly growing digital content business and decreasing business with optical media storages such as CD and DVD urged the development of Sony DADCs PA System. Since the system was available in 2006, the number of product activations increased daily and all major customers of SecuROM^{TM}started to publish their content online. This development in software business also required - and still requires - ongoing development of media independent DRM solutions such as SecuROM^{TM}PA.
Almost every time a customer request causes the development of new features, the implementation requires an update of the existing replication system. Several circumstances made it impossible for the development team to maintain the replication system and it could not be launched. This project has been initiated to get the Active/Active Replication up and running again but also to enhance the systems performance.

1.2 Ambition and Structure

Before the actual realization of the project, it is necessary to become familiar with the subject of Active/Active Replication to get an understanding of the requirements. Even though parts of the system already exist, it is absolutely important to understand the theories and concepts as to comprehend the actual implementation.

For the realization of the project a determination and analysis of the existing implementation is required:

- Determination of replication system status
- Definition of required updates and implementations

Another milestone is the implementation of changes and updates:

- Encapsulation of Synchronization and Replication
- Update and adaption of existing source code
- Implementation of missing replication system parts

The backoffice is updated for administrative purposes:

- Implementation of an interface to kill running threads

- Implementation of an interface to select data for replication grouped by Data Centers
- Implementation of a monitoring tool for replication jobs

During the implementation, test cases are created :

- Examination and update of existing functionality- and load- tests
- Implementation of new functionality- and load- tests
- Detection and reduction of replication latency

The final implementation, in combination with the existing infrastructure, guarantees high availability. Moreover, permanent data consistency via the Active/Active Replication system ensures the reliability of Sony DADCs DRM Product Activation Servers.

1.3 Definition of Active/Active Replication

High availability is an essential requirement for productive systems of any kind. Customers expect services to be up and running at any time without flaws. As a matter of fact, no system is perfect and may fail at some point.
The brilliance of Active/Active Replication systems is that even if an error occurs, a fix is provided fast. If the time between the appearance of an error and its fix is not noticeable, availability is maintained.
Respectively it is necessary to have an infrastructure in place, where services are available at independent destinations to prevent failure due to catastrophes. Beside service availability, infrastructure needs to assure data integrity and consistency. Any changes to relevant data of one system needs to be broadcast and applied to any other replicated system. If one instance of the system fails, another instance has to take over within a split second, delivering the same result as the former instance would have delivered.

1.3.1 Active/Active Replication Techniques

There is not only one way how an Active/Active Replication can be implemented. Depending on the purpose of the application, it is necessary to find the one technique that fits best for the given requirements.

1.3.1.1 Asynchronous Replication

For asynchronous replication, the system requires some kind of synchronization queue to track changes. Any modification on relevant data creates an entry within this queue. The modification information is then sent to any other instance of the system. Each destination receives a notification and performs the necessary updates.
The replication does not influence the main functionality of the application and could even be implemented completely independent. Due to this circumstance the performance of the application is not affected by the replication. However, the downside of asynchronous replication is *replication latency*.
Replication latency is the time delay between the change on the source database and the update on the destination database. This delay leads to temporary data inconsistency and could even lead to data collision and permanent data inconsistency.
Imagine the same data is modified on two different instances of the replication system at the same time. Both instances will enqueue a change entry and send it to the each other. Due to replication latency, either instance is unaware of the other instances change at this time. When the update is finally applied to the respective destination, the original change is overwritten and the data is not consistent anymore. [1]

1.3.1.2 Synchronous Replication

To prevent data inconsistency, synchronous replication needs to be implemented. Basically synchronous replication follows the same approach as asynchronous replication. Whenever data is modified, an entry is written to some synchronization queue and sent to any other instance of the system. In case of synchronous replication, changes are not committed at this stage but the sent data is only used to lock affected data sets.
This step is followed by a check to verify if all destinations could successfully lock the required data. Should the verification be positive, the source, as well as the destinations, commit the update. Otherwise the update is rejected and some error service routine is executed. This concept is called two-phase commit protocol and commonly used by transaction managers [2].
Even though synchronous replication ensures synchronism of data, there is a performance disadvantage for the application. Data modifications can only be performed when all instances are ready to commit the update. The resulting delay is called *application latency.* [1]

1.3.2 Initial Replication Implementation

For the purpose of Sony DADCs PA System, asynchronous replication has been the preferred choice as replication latency is less of an issue than application latency.
Furthermore the majority of data modifications are triggered due to software application activations which are performed by end users. Aforesaid are not supposed to experience any unacceptable delays during the activation process. A lock on data items due to activations would permanently increase the application delay as the amount of activations is on a consistent high level. Apart from that, modifications by end user related events are not extremely consistency critical.
In comparison to those, administrative modifications on the database, which are consistency critical, are performed on a manageable basis. Thus critical inconsistency conflicts are not to be expected.
In summary it can be said that as for these reasons, the PA Systems fulfills the requirements for asynchronous replication referring to the following quotation [2, Page 136]:

> If data collisions are not deemed to be a serious problem, then both systems can be actively processing all transactions;
> ...
> Alternatively, if the cost of resolving data collisions is small, or if data collisions can be resolved automatically, asynchronous Active/Active Replication may also be appropriate.

1.3.3 Symbolic Illustration

Figure 1.1 is a symbolic illustration of the required failure-resident infrastructure. Several Data Centers are located at independent destinations all over the world. Each Data Center (DC) may exist of n application servers (PA Server XX[2]#[3]) but only *one* shared database (Database XX). DCs are interlinked via administrative configuration. Once configured, the Active/Active Replication is enabled.
When a DC receives a request, a load balancer distributes it to any PA Server and the request is processed. If a relevant database modification is caused, the corresponding PA Server creates a synchronization queue entry (see subsubsection 1.3.1.1) for each

[2]indicates the server location
[3]indicates the servers identification number

known DC.
To avoid replication conflicts, the active replication is triggered by only one Master PA Server (highlighted in orange) within each DC. Never the less, each application server has the ability to perform the active replication in case the actual Master server is not available anymore. In such a case, the one PA Server within the DC which has the lowest identification number becomes the new Master and performs further active replication actions.
For the actual replication sequence flow, see Figure 2.4

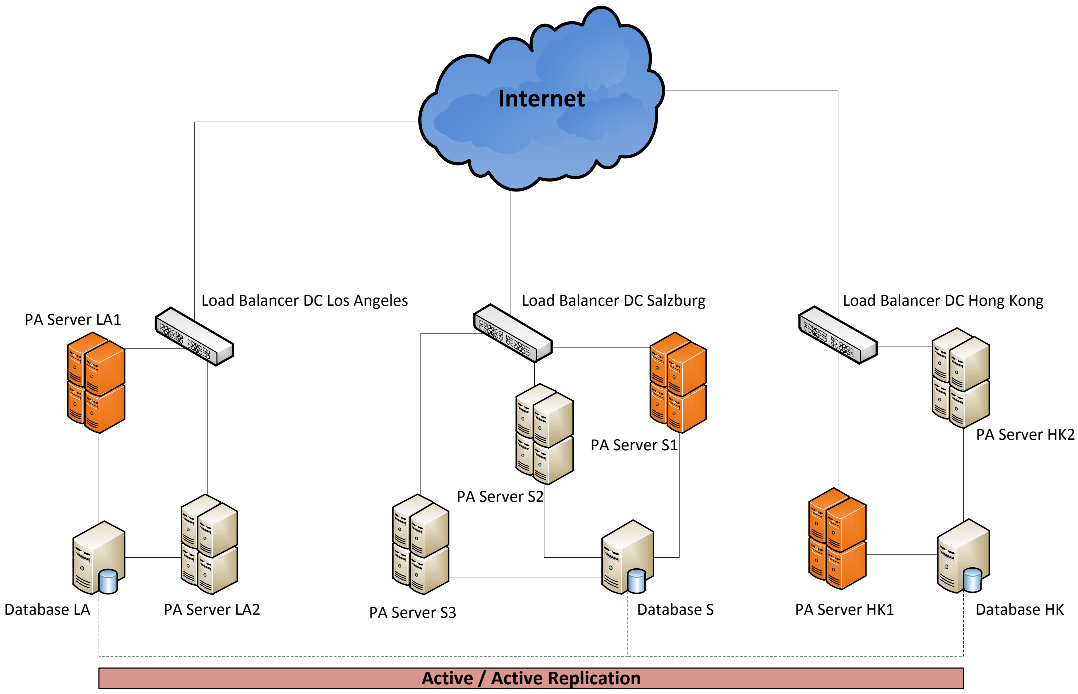

Figure 1.1: Overview SecuROM^{TM}Product Activation Server Architecture

2

Analysis of the Existing System

2.1 Database Scheme

Figure 2.1 shows a high level overview of the PA Systems database. In combination with the following information, it is easier to get an idea of the PA System.
The majority of entities have to be replicated, but some are either used for local server settings or part of the replication and must not be replicated.
These exceptions cover the entities within the orange box in the upper right of the illustration as well as the *USERLOGGING* entity.
As for all other entities the replication

- has already been implemented and works, or
- has been implemented but needs to be adapted, or
- is missing and needs to be implemented completely.

An analysis of the replication status at project launch has been done and can be found under section 2.2.

To understand the PA Server Scheme the following sections describe the relations between entities and users of the database. Each user can be considered as operator layer, starting from the highest - the end user - to the lowest layer - the application.

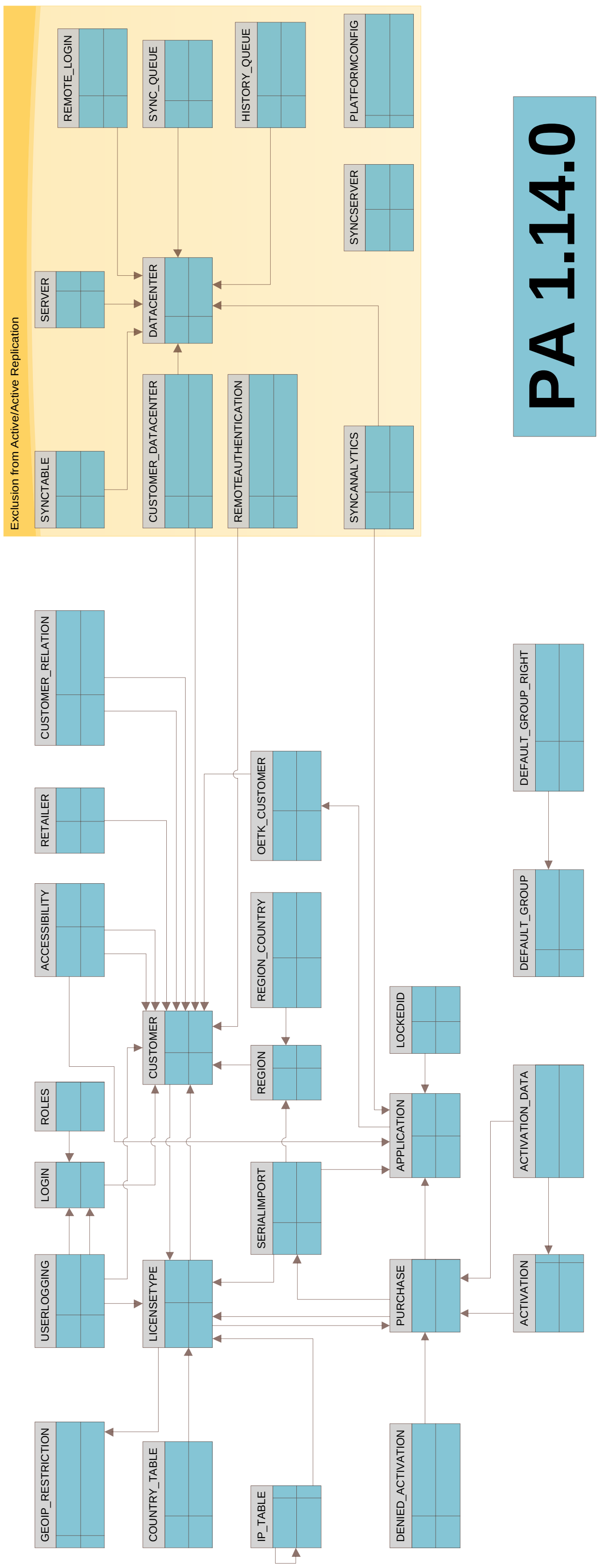

Figure 2.1: Overview SecuROM™Product Activation Server Scheme [3]

2.1.1 End User

2.1.1.1 End User Interaction

The PA Systems main purpose is the deployment of a service that allows users of software applications to activate digitally purchased content by using a dedicated interface. In most cases the interface is an adoption of SecuROM^{TM}'s standard Product Activation Unlock Library (PAUL) which basically displays dialog windows, guiding end users step by step through the activation process (see Figure 2.2).

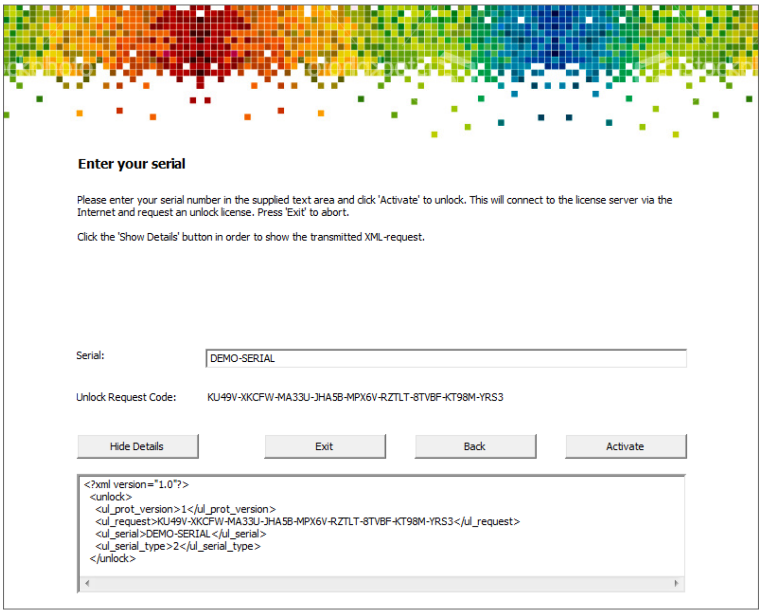

Figure 2.2: SecuROM^{TM}Standard Product Activation Unlock Library Dialog

As illustrated in the lower part of Figure 2.2, the library prepares an Extensible Markup Language (XML) statement which is sent to the PA Server. The server then processes the XMLs content, checks for relevant database information and generates an XML reply to either unlock the application or display any other information.

Beside the activation process, there could be further communication between the PAUL and the PA Server to handle various other tasks. There may be revocations, registrations and so forth. All communication is based upon XML with the purpose of helping end customers to handle their digitally purchased content.

These events cover the majority of database entry modifications, as mentioned under subsection 1.3.2, and need to be processed rapidly. For these modifications, performance is more important than data consistency although it is important to replicate changes to all DCs of course.

2.1.1.2 Relevant Entities for End Users

Table 2.1 indicates entities that may change due to end user actions

Entity	Description
PURCHASE	Single purchase data
ACTIVATION	General activation data
ACTIVATION_DATA	Specific activation data and link to purchase

Table 2.1: Relevant Entities for End Users

2.1.2 Customer

2.1.2.1 Customer Interaction

Before any activation process can be performed, data, such as serial numbers and license settings, has to be available within the database. Via the PA Servers backoffice system (see Figure 2.3), customers have the ability to administrate this data.

Figure 2.3: SecuROM™Product Activation Server Backoffice

In case of data changes via the backoffice, it is important that all modifications are replicated to other DCs immediately. Given the fact that changes only happen once in a while, data inconsistency is not to be expected.

2.1.2.2 Relevant Entities for Customers

Table 2.2 indicates entities that may change due to customer actions

Entity	Description
SERIALIMPORT	Container for purchase batches
PURCHASE	See Table 2.1
REGION	Regions for analytics purposes
RETAILER	Retailers for analytics purposes
ACTIVATION	See Table 2.1
APPLICATION	Application data and link to customer
ACCESSIBILITY	Customers access rights for applications
LICENSETYPE	License data for purchases
GEOIP_RESTRICTION	GeoIP restriction data for countries
COUNTRY_TABLE	Country based activation start date information
IP_TABLE	Internet Protocol (IP) range based activation allowances

Table 2.2: Relevant Entities for Customers

2.1.3 Administrator

2.1.3.1 Administrator Interaction

PA Server administrators are in charge of the servers system configuration. Part of the configuration is the basic user setup, which is required for any customer to get access to the backoffice, moreover to perform any actions. Administrative operations, similar to customer operations, are handled via the backoffice and as for all parties, it is important that relevant changes are replicated.

2.1.3.2 Relevant Entities for Administrators

Table 2.3 indicates entities that may change due to administrative actions

Entity	Description
SERIALIMPORT	See Table 2.2
PURCHASE	See Table 2.1
REGION	See Table 2.2
RETAILER	See Table 2.2
ACTIVATION	See Table 2.1
APPLICATION	See Table 2.2
LOCKED_ID	Locked applications
LICENSETYPE	See Table 2.2
GEOIP_RESTRICTION	See Table 2.2
COUNTRY_TABLE	See Table 2.2
IP_TABLE	See Table 2.2
CUSTOMER	Customer data
CUSTOMER_RELATION	Customer relation information
LOGIN	User account data and customer relations
ROLES	User account role informations
ACCESSIBILITY	Table 2.2
OETK_CUSTOMER	PA Server / Online Encryption Tool Kit (OETK) Server relation information
DEFAULT_GROUP	Default role groups
DEFAULT_GROUP_RIGHT	Specific role data for groups

Table 2.3: Relevant Entities for Administrators

2.1.4 Application

2.1.4.1 Application Interaction

Of course, the lowest layer that is required by administrators is the PA Application itself and even the application may perform changes that need to be replicated.

2.1.4.2 Relevant Entities for the Application

Table 2.4 indicates entities that may change due to actions performed by the PA Application

Entity	Description
USERLOGGING	User login logging data

Table 2.4: Relevant Entities for the Application

2.2 Initial Replication Status

The initial status of the existing replication implementation was unknown, therefore it is necessary to perform several tests to find out, which parts of the database replication implementation were already covered, partial covered or missing entirely.
Due to the systems complexity it is important to test various possibilities of data modification to get a real snapshot of the initial status.
As those tests are not only required for an analysis of the initial implementation, but also for later testing purposes and functionality assurance, reusable tests are created via JMeter[1].
Not all tests could be automated yet, and some have been performed manually. However, the results are listed in Table 2.5.

Table	**Working**	**Not Working**	**Missing**
ACCESSIBILITY	•		
ACTIVATION		•	
ACTIVATION_DATA		•	
APPLICATION	•		
COUNTRY_TABLE			•
CUSTOMER	•		
CUSTOMER_RELATION	•		
DEFAULT_GROUP	•		
DEFAULT_GROUP_RIGHT	•		
DENIED_ACTIVATION			•
GEOIP_RESTRICTION	•		
IP_TABLE			•
LICENSETYPE	•		
LOCKEDID			•
LOGIN	•		
OETK_CUSTOMER	•		
PURCHASE	•		
REGION			•
REGION_COUNTRY			•
RETAILER			•
ROLES	•		
SERIALIMPORT		•	
SERIALIMPORT_COUNTRY			•

Table 2.5: Replication Status

[1] See `http://jakarta.apache.org/jmeter/`

2.3 Initial Replication Implementation

2.3.1 Server Technologies

The PA System runs a JBoss Application Server (JBoss AS)[4]:
A Java EE[2] certified platform for developing and deploying enterprise Java applications, Web applications, and Portals, JBoss Application Server provides the full range of Java EE 5 features as well as extended enterprise services including clustering, caching, and persistence.
Therefore technologies such as Java Servlets, Enterprise JavaBeans and Java Management Extensions (JMX) are part of the applications concept.

- Java Servlets [5]
  A servlet is a Java programming language class used to extend the capabilities of servers that host applications accessed via a request-response programming model. Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by Web servers. For such applications, Java Servlet technology defines HTTP-specific servlet classes.
- Enterprise JavaBeans [6]
  Enterprise JavaBeans (EJB) technology is the server-side component architecture for Java Platform, Enterprise Edition (Java EE). EJB technology enables rapid and simplified development of distributed, transactional, secure and portable applications based on Java technology.
- Java Management Extensions [7]
  The JMX technology provides the tools for building distributed, Web-based, modular and dynamic solutions for managing and monitoring devices, applications, and service-driven networks. By design, this standard is suitable for adapting legacy systems, implementing new management and monitoring solutions, and plugging into those of the future.

2.3.2 Active/Active Replication

The main part of the initial implementation was a combination of a previously implemented synchronization mechanism for another service and the actual replication

[2] Java Platform, Enterprise Edition (Java EE)

system. Even though the synchronization mechanism and the replication system are completely independent from each other, they were not separated. This condition only led to confusion and a new structure is required. Most certainly an encapsulation of the routines must be done. An implementation of both routines without dependencies has to be established. One with the purpose of synchronizing OETK application data with the PA System, the other one to perform the Active/Active Replication.

Figure 2.4 shows the replication sequence flow. What looks pretty straight forward in theory is quite complex in praxis and changes need to be done with great care. Common pitfalls are missing dependency information which, if not fixed, lead to null pointer exceptions. Another source for errors are missing updates for Structured Query Language (SQL) statements.
Besides corrupted and non updated source code, bad implementation makes it harder to read the code and, even worse, does not contribute to the applications performance.

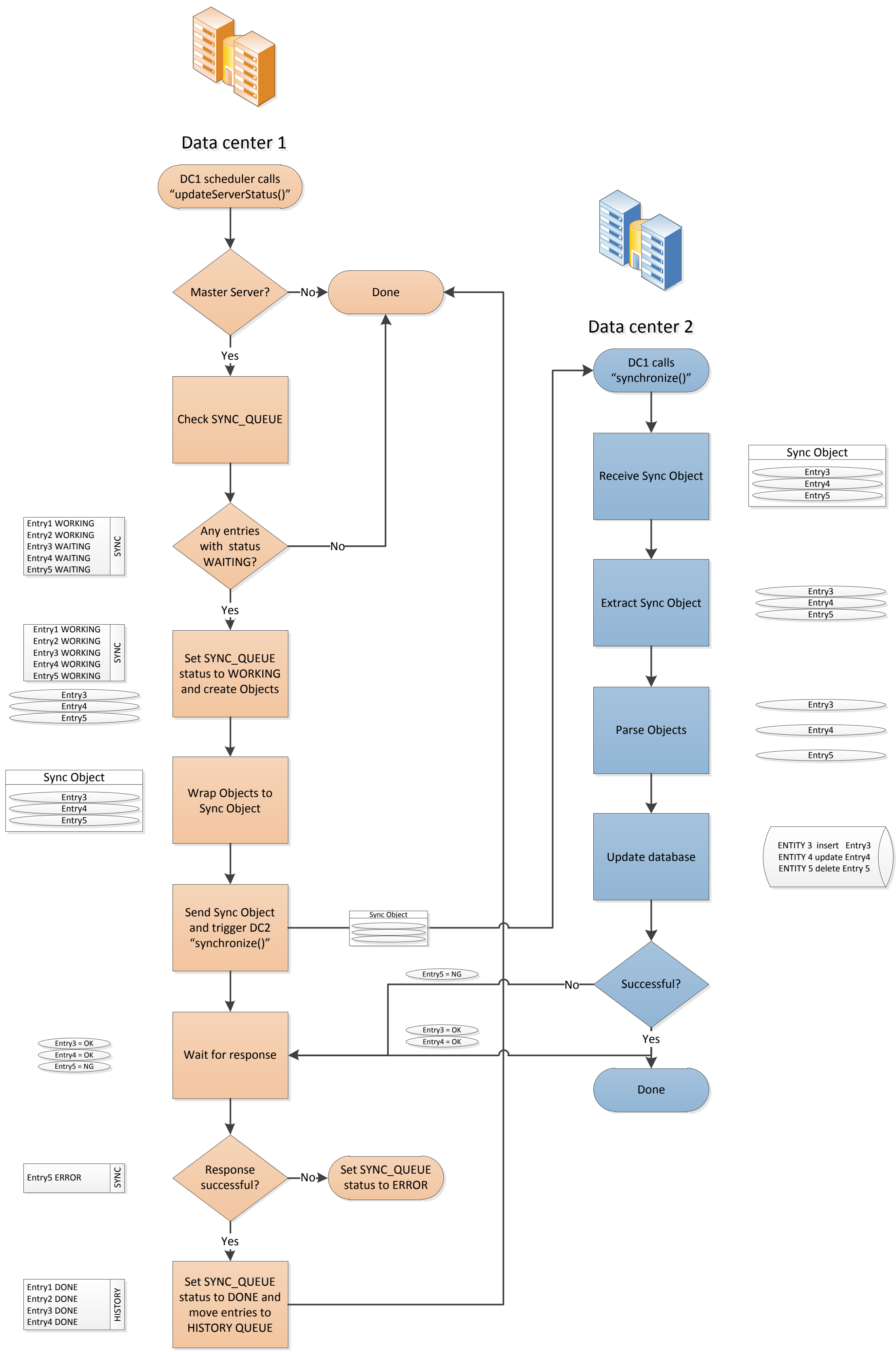

Figure 2.4: Replication Sequence Flow Chart

3

Development

3.1 Encapsulation of Synchronization and Replication

To refer to subsection 2.3.2 the former implementation was a mix of an independent synchronization mechanism and the replication system. Encapsulating the routines simplifies further development and makes changes less prone to error. Two independent instances are essential for the codes separation. Shared methods need to be identified and decoupled. Permanent testing and debugging is required to detect and resolve eventual problems like null pointer exceptions.

The encapsulation involves radical changes of the application and a new branch is created. Therefore any changes must be noted and replicable for a transparent transfer to the actual development build.

As the application exists of many classes with dependencies to either the synchronization mechanism or the replication system, lots of source code updates are required. However, updates within existing source code implies regular testing to ensure that the system still works. This is especially important for methods where there is uncertainty of their exact use.

Actually, the encapsulation turned out to be the most challenging part of the project. Interdependencies are hard to predict and tests have to be done bi-directional - appropriate to Active/Active Replication systems. Due to the applications complexity, even simple changes could have great impact on the application and structural changes are only partial possible.

3.2 Update of Corrupted Source Code

Some existing parts of the replication were identified as non working. Detailed analysis and debugging showed that SQL statements were not adjusted to database changes. In some cases bad implementation is the source of error.
While analyzing the source code, several quick fixes could be applied, resulting in optimization of the replications performance. By improving the performance, replication latency could be reduced. This reduction also means the updated system is less prone to data inconsistencies and collisions.
Further improvements are possible and will be focused for the ongoing development.

3.3 Implementation of Missing Source Code

To achieve the objective, all required entities have to be part of Active/Active Replication. One major disadvantage of the implementation is the fact that database updates, especially the introduction of new entities, do not dynamically slip into the replication. Actually the opposite is the case, and new entities mostly require new attributes within the replications SYNC- and HISTORY_QUEUE. Thus, implementing the replication for new entities goes hand in hand with high efforts.
Including a new entity issues the update of almost the entire replication system. Only few methods remain unchanged and one has to work with great care to ensure that complete functionality is given. Therefore updates of existing tests are required to verify no information is lost during replication.

4

Conclusion

With the growing number of end-users who have to rely on Sony DADCs PA System, it is getting more important than ever before to establish a working Active/Active Replication system. The current implementation is almost ready for final Quality Assurance (QA) tests and in case the QA result is positive it could even be used for the live system.
However, the original development concept comprises major design faults. Even though it is working, it is almost impossible to maintain the system; especially as several developers are working on features for the application and implement database updates. Of course, not everything needs to be redone, but modifications to allow an easy and dynamic way of adding entities or attributes should be considered.
Fundamental decisions, such as using asynchronous Active/Active Replication have definitely been well-wrought, so there is no need to raise a doubt. Once the replication system is enabled a good basis for availability and reliability of the PA System is set.

4.1 Project Summary

The project has been launched with the set up of the development environment. Next to easily accessible tools such as NetBeans, ORACLE SQL Developer et cetera, access to Apache Subversion (SVN) and the PA Systems staging environment were necessary. This set up could only be performed at Sony DADC. To use these company internal systems from an external access point, a Virtual Private Network (VPN) connection needs to be established.
Most of the project work requires knowledge in Java development, which has already been trained in Austria. Another prerequisite is at least basic knowledge about

database systems. Therefore a database class has been attended at the Southern Utah University.
Through an unfortunate accident during the semester, the operating system broke. After a fresh operating system set up, it was not possible to establish a connection to Sony DADC Austria's internal network anymore. Even though Sony DADCs IT department tried to help with a remote set up, the system could only be fixed back in Austria, in middle of January. When the accident happened the projects progress has been rather small and most of the project work had to be done within the last two month in Austria.

Working on a complex application like the PA System, where many interdependencies have to be incorporated, made it hard to fulfill all objectives. Therefore, it was not possible to cover the whole project scope and some of the requirements (see section 1.2) are still under construction.

4.2 Perspective

The project is planed to be completed in course of a Bachelor Thesis. The idea is to take advantage of noticed flaws and design a proper development concept based on dynamic design patterns. As there is a parallel development project with the scope to update the current JBoss AS version, latest development technologies will soon be supported. Chances are that this update will not only allow the use of new development concepts but it will also reduce complexity and simplify coding.

Bibliography

[1] Sombers Associates Inc. and W. H. Highleyman. What is active/active? Internet, October 2006.

[2] B. Highleyman. *Breaking the Availability Barrier: Survivable Systems for Enterprise Computing*. Authorhouse, 2003.

[3] Michael Auracher. Pa 1.14.0. Sony DADC Austria AG internal document, 12 2010.

[4] Jboss application server. World Wide Web. Accessed March, 2011.

[5] What is a servlet? World Wide Web. Accessed March, 2011.

[6] Enterprise javabeans technology. World Wide Web. Accessed March, 2011.

[7] Java management extensions (jmx) technology. World Wide Web. Accessed March, 2011.

List of Abbreviations

DADC Digital Audio Disc Corporation

DRM Digital Rights Management

PA Product Activation

PAUL Product Activation Unlock Library

IT Information Technology

DC Data Center

XML Extensible Markup Language

IP Internet Protocol

OETK Online Encryption Tool Kit

Java EE Java Platform, Enterprise Edition

JBoss AS JBoss Application Server

JS Java Servlet

EJB Enterprise JavaBean

JMX Java Management Extensions

MBean Managed Bean

SQL Structured Query Language

QA Quality Assurance

SVN Apache Subversion

VPN Virtual Private Network