UNIVERSITY OF ALASKA ANCHORAGE

CSCE A470

CAPSTONE PROJECT

SQL Server Failover Effects on Applications Connected to the Cluster

Author:

James A. Tweet

Supervisor

Prof. Mock, PhD

Anchorage AK, May 2016

Computer Science & Engineering Department

UNIVERSITY of ALASKA ANCHORAGE

Table of Contents

List of Figures

Abstract

In this project I will build a SQL Server failover cluster and examine what happens to programs that are connected to a SQL Server cluster during a failover. Then try to mitigate the effects of the failover on the software. Finally pass a SQL Server certification test from Microsoft.

Chapter 1
Introduction

1.1 The SQL Server failover cluster

This chapter will discuss the SQL Server failover cluster, database connections, motivations, and the plan for the project.

A SQL Server failover cluster requires at least two computers which can access the same set of hard drives. Generally this is accomplished by having two servers connected to the same SCSI array.

One of the drives is the quorum disk. The cluster uses the quorum disk to keep track of which server is active. The rest of the disks are for the databases, log files, etc. in SQL Server.

In a failover cluster there will be one active server and a passive server. If the active server goes down for some reason then the passive server becomes active. This is called a failover event.

Figure 1 is an example of two-node failover cluster configuration: Two servers, each with two HBAs and four storage enclosures (Gregory Kincade, 2015). The entire purpose of a SQL Server failover cluster is high availability. You want to ensure if any component fails the system will continue to operate. Having the server down can become very expensive fast for example if you have 1000 employees paid an average of $15/hr. If the server is down for an hour the company is out $15,000 just in wages.

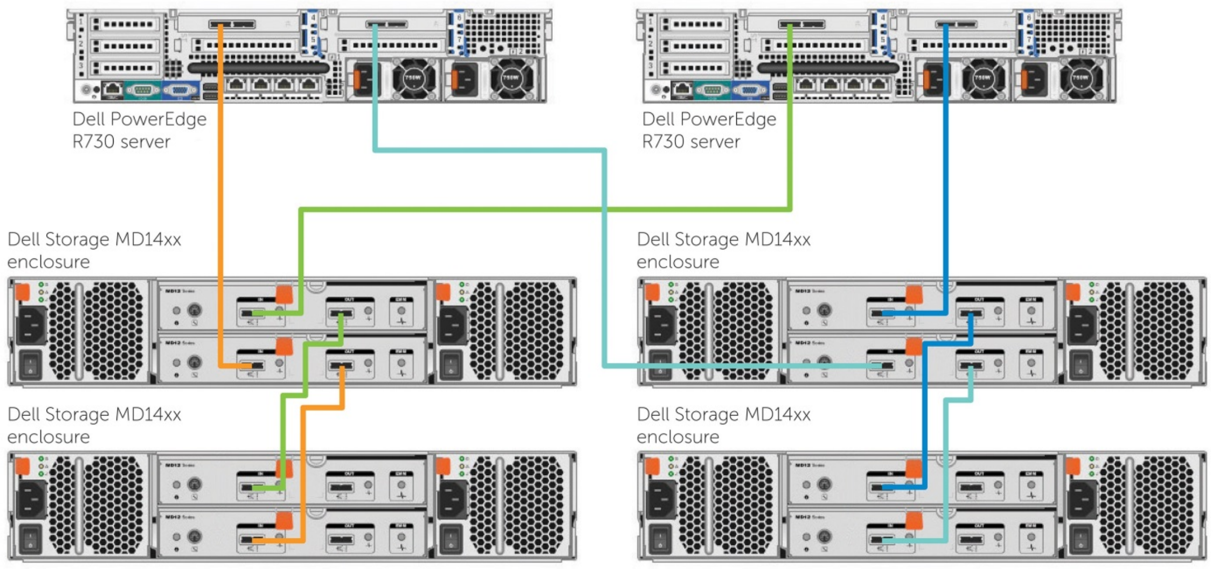

Dell PowerEdge R730 server
Dell PowerEdge R730 server
Dell Storage MD14xx enclosure
Dell Storage MD14xx enclosure
Dell Storage MD14xx enclosure
Dell Storage MD14xx enclosure

Figure 1-1 Dell SQL Server Cluster

1.2 ADO.NET, ODBC, JDBC

For this project I will study three different database connection methods. With all three connections there are a few basic steps that they all follow.

1. Connect to database
2. Fetch a record set
3. Add, update or delete one or more records
4. Save the changes in the record set back to the database
5. Close the record set
6. Close the database connection

Between each of those steps I will failover the cluster. This project is all about what happens to the program after the failover.

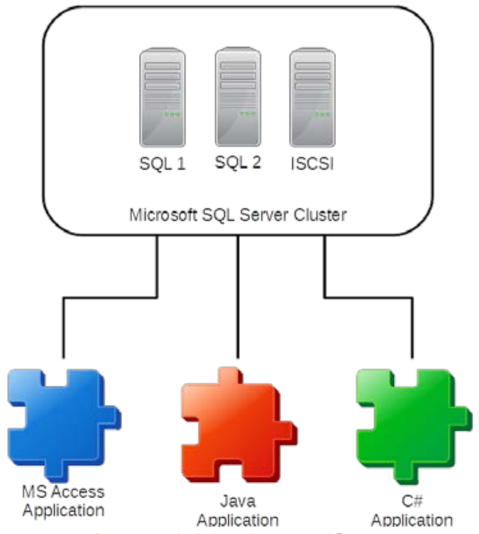


Figure 1-2 SQL Server Cluster

Microsoft Access uses an ODBC connection to the database. The C# application uses ADO.NET connection to the database. Finally the Java application will use JDBC.

1.3 Learning from the past

A number of years ago I worked for an access control company. The main software that we sold used SQL Server to store records (Identiv, 2015). A few of our customers had us use a SQL Server failover cluster to store records. The software worked fine until a failover event occurred.

If a failover event occurred all of the workstations would need to restart the software. I would like my software to be more robust to handle a failover event automatically.

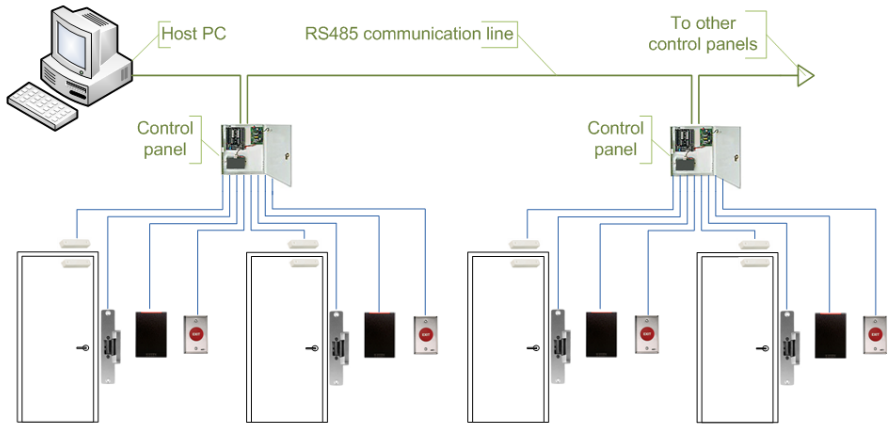


Figure 1-3 Example Access Control System (Wikipedia, 2016)

Also over the years I have created quite a number of MS Access databases that connect to SQL Server. Most of these databases have just a few people who enter data. Then the manager would use the system to get a monthly report. Now I would like to see what happens if the SQL cluster failover causes problems in MS Access.

1.4 The Plan

First I need to have a SQL Server failover cluster to run my tests. I will setup three virtual PCs as the cluster. The first two PCs will run Windows 2008 Server software. The third PC will run Linux with the ISCSI target software (Openfiler, 2015). Next I will setup the cluster between the two Windows servers. Then SQL Server 2014 will be installed onto the cluster. Along with the servers for the cluster I will need a domain controller running Windows 2012 and a client running Windows 8.1

Once all of the software has been installed I will use SQL Server Management Studio to connect to the SQL Server and add three databases. Next I will add a table and some records to each database.

Now that I have verified that SQL Server is operating properly I will test the failover cluster. I will cause a failover event to occur then make sure that SQL Server is still working properly.

Now I will build a Microsoft Access database that connects to the SQL Server. This database will be tested to ensure that it can add, update and delete records. Next I'll build a C# program that connects to the cluster. This C# program must also add and update records. Finally I'll build a Java program that connects to the cluster which will also add and update records.

These programs that I have created will now be tested to see the effects of a failover event. First the program will connect to the database. Then I'll failover the cluster. Next check to see if the connection is still valid if the connection is lost reconnect. Next the program will open up a record set. Then I'll failover the cluster again. Now test the record set to see if I can iterate the records and close the set. The next test will be to open a record set, add a record then failover the cluster. Now see if the added record can be saved to the database. This test will happen the same way for updating and deleting records.

Now that I know the effects of a failover event can I protect the user from these effects? Can I get an event to occur when the failover occurs? Does just reconnecting the database connection solve the problem? Do I need to reopen the record set? To answer these questions I will modify the three programs to attempt to mitigate the effects of a failover event. As I modify the software I will continue to test if the modifications worked as expected.

Once I am satisfied that I have made all of the possible modifications I'll retest all of the failover events. Then document the modifications and the results. It is possible that MS Access is unable to recover from a failover.

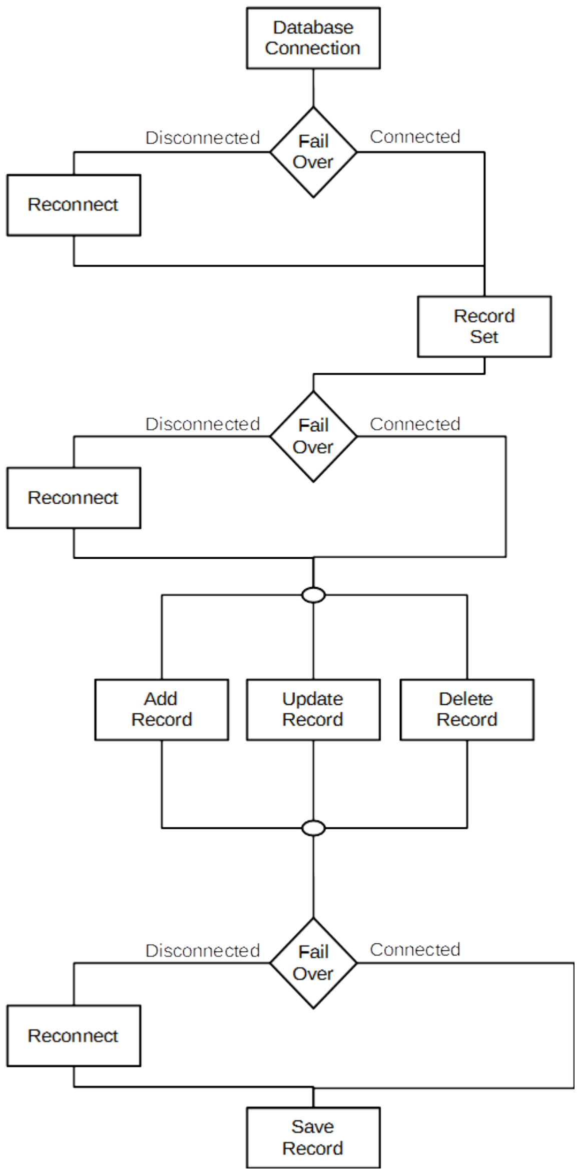


Figure 1-4 Testing Flow Chart

Microsoft® SQL Server®

Figure 1-5 SQL Server Logo

The final part of my project is to pass Exam 70-462 Administering Microsoft SQL Server 2012 Databases (Microsoft, 2012). Part of this exam covers setting up a SQL Server cluster.

Chapter 2

2.1 Technologies

This chapter will discuss the technologies that will be used in my project. Also it will discuss the design and timeline of the project. This project will use the following technologies.

- Oracle VirtualBox
- Microsoft Windows Server
- Microsoft Cluster Services
- FreeNAS
- Microsoft SQL Server
- C#
- Java
- Microsoft Access

Oracle VirtualBox

Figure 2-1 Oracle VirtualBox Logo

VirtualBox is a cross-platform virtualization application. What does that mean? For one thing, it installs on your existing Intel or AMD-based computers, whether they are running Windows, Mac, Linux or Solaris operating systems. Secondly, it extends the capabilities of your existing computer so that it can run multiple operating systems (inside multiple virtual machines) at the same time. So, for example, you can run Windows and Linux on your Mac, run Windows Server 2008 on your Linux server, run Linux on your Windows PC, and so on, all alongside your existing applications. You can install and run as many virtual machines as you like -- the only practical limits are disk space and memory. (Oracle Corporation, 2016)

Microsoft Windows Server

Microsoft Windows Server is the main operating system that will be used for this project. Windows Server controls the user authentication and runs the cluster services. *Windows Server is the platform for building an infrastructure of connected applications, networks, and web services, from the workgroup to the data center.* (Microsoft, 2016)

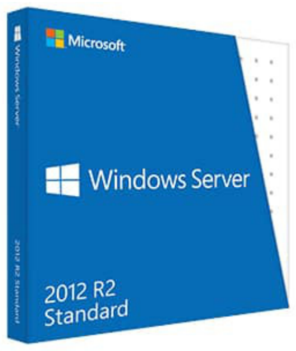


Figure 2-2 Microsoft Server 2012 R2

Microsoft Cluster Services

Microsoft Cluster Server (MSCS) is computer program that allows server computers to work together as a computer cluster, to provide failover and increased availability of applications, or parallel calculating power in case of high-performance computing (HPC) clusters (as in supercomputing).

Microsoft has three technologies for clustering: Microsoft Cluster Service (MSCS), Component Load Balancing (CLB) (part of Application Center 2000), and Network Load Balancing Services (NLB). In Windows Server 2008 and Windows Server 2008 R2 the MSCS service has been renamed to Windows Server Failover Clustering and the Component Load Balancing (CLB) feature has been deprecated. (Wikipedia, 2015)

The Microsoft Cluster Services allow critical services to be automatically moved to a secondary server if the primary server fails.

Microsoft SQL Server

Microsoft SQL Server is a relational database management system that can be run on a Windows Server. SQL Server is the database engine you connect to access information stored in the database.

SQL Server 2014, like its predecessors, is more than a database engine. It is a collection of components that you can implement separately or as a group to form a scalable, cloud-ready information platform. In broad terms, this platform is designed for two purposes: to help you manage data and to help you deliver business intelligence. (Microsoft, 2014)

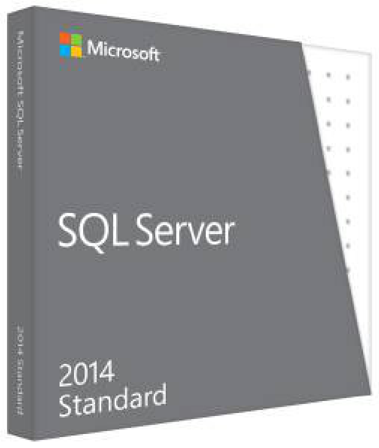


Figure 2-3 SQL Server 2014

Microsoft C#

C# is an elegant and type-safe object-oriented language that enables developers to build a variety of secure and robust applications that run on the .NET Framework. You can use C# to create Windows client applications, XML Web services, distributed components, client-server applications, database applications, and much, much more. Visual C# provides an advanced code editor, convenient user interface designers, integrated debugger, and many other tools to make it easier to develop applications based on the C# language and the .NET Framework. (Microsoft, 2016)

Figure 2-4 Microsoft Visual C#

Java

The Java® programming language is a general-purpose, concurrent, class based, object-oriented language. It is designed to be simple enough that many programmers can achieve fluency in the language. The Java programming language is related to C and C++ but is organized rather differently, with a number of aspects of C and C++ omitted and a few ideas from other languages included. It is intended to be a production language. (Gosling, Joy, Steele, Bracha, & Buckley, 2015)

Figure 2-5 Java

MS Access

Microsoft Access, often abbreviated "MS Access," is a popular database application for Windows. Access allows users to create custom databases that store information in an organized structure. The program also provides a visual interface for creating custom forms, tables, and SQL queries. Data can be entered into an Access database using either visual forms or a basic spreadsheet interface. The information stored within an Access database can be browsed, searched, and accessed from other programs, including Web services.

While Access is a proprietary database management system (DBMS), it is compatible with other database programs since it supports Open Database Connectivity (ODBC). This allows data to be sent to and from other database programs, such as MS SQL, FoxPro, Filemaker Pro, and Oracle databases. (PC.net, 2016)

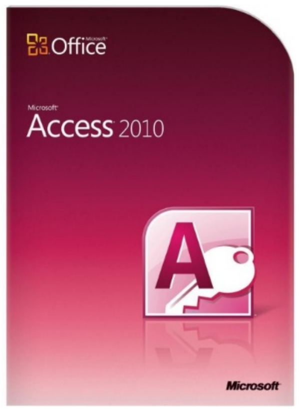


Figure 2-6 Microsoft Access

2.2 Design

To run the tests I want I will need to have a domain controller, DNS, ISCSI target, two failover cluster nodes and a client PC.

Domain Controller Computer

The domain controller is its own virtual computer running Windows Server 2012 R2. On this machine Active Directory and DNS services will be installed. A local domain called Test.UAA is configured in Active Directory. The virtual computer is configured with 2 GB ram, 2 processors, and a network card. Originally the hard drive was 25 GB but Windows needed more space so I increased the drive to 75 GB. Once all the services were installed and all the updates complete the system actually needed 30 GB.

ISCSI Target

The ISCSI Target is its own virtual computer running FreeNAS. Originally I was going to run OpenFiler as the ISCSI target but it turns out that OpenFiler does not support the SCSI 3 protocol. This means that it will not work for a failover cluster. It will work for a cluster but not a failover cluster. In a failover cluster one server has exclusive access to the ISCSI target drive. This can only be done using the SCSI 3 protocol. Whereas a cluster uses the ISCSI target like a network share and many computers can access the drive at the same time.

The virtual PC running FreeNAS had 2 GB of ram, 2 processors, and a network card. On this virtual PC there were 3 hard drives. The first hard drive was 12 GB and contained the operating system. Only 650 MB was used of the available space. The second drive was 4 GB. This was the quorum drive and only used 319 MB of the available space. The third drive was 16 GB but only used 1.35 GB of space. This was the data drive that contained the SQL Server databases.

SQL1 & SQL2 Computers

The two failover node computers had to be exactly the same. You can’t have one node running Windows Server 2012 and the other node running Window Server 2012 R2. The operating system and the hardware must to be identical. When you install the cluster services the install software will even check to see if you have the same updates on each system.

Both virtual PCs have 4 GB ram, 2 processors, a 75 GB hard drive and 2 network cards. Originally I was going to have one of the PCs be the domain controller along with being the failover node. Unfortunately both nodes must be configured the same. It may be possible to have both PCs domain controllers and failover nodes. I did some research on the internet regarding setting up domain controllers on the failover nodes and generally people would say that this is a bad idea. So it is much easier to setup a separate virtual PC as the domain controller.

You’ll notice that the failover nodes have 2 network interface cards unlike all of the other PCs. One network card is connected to the same network as all of the other PCs. The second network card is used for internal communication between the nodes. In a real setup with actual servers the second network cards are connected directly together with a crossover network cable.

On both nodes I installed Windows Server 2012 R2 and setup failover clustering. I also installed the VirtualBox Guest Additions, which give you the ability to share the clipboard, drag and drop files between the host and virtual PC and adjust the screen size. Then I tried to install SQL

Server onto the cluster. Once I got to the screen where I would enter the name for the SQL Server and instance name, I would get an error when I clicked on the next button.

The error I received was that SQL Server could not determine if the name I had chosen already exists on the network. After some searching on the internet regarding this error message I found out that to trouble shoot this farther I would need the exact error number from the API call that SQL installation used. The API call is NetServerGetInfo. From Microsoft you can download some C source code that would call this API for a server name you add as an argument. So from the command line I run the executable created from the C code. I look up the error code and it says that the call can't determine if the host name exists. Thanks Microsoft, you are so helpful. From the console I try to ping the host name and the DNS look up does not find that name. I decide to start uninstalling things that may be unnecessary. So I uninstall the guest additions and that does the trick. The API call starts working as expected. Now I can continue past the name screen on the SQL Server installation but my troubles aren't over.

Originally the plan was to have the domain controller and both failover nodes run Windows Server 2012 R2 as the operating system. This worked well until I tried to install SQL Server 2014 onto the failover cluster. At some point during the install I would get an error implying that the logins I used for the SQL Server services did not have the correct permissions. The installation would complete but SQL Server was not properly on the cluster. At this point if you uninstalled SQL Server and tried to reinstall it, the reinstall would fail. To get rid of SQL Server you had to reinstall Windows.

According to Microsoft, the user credentials for SQL Server services need to be a domain user and logon as a service permissions. I did not give logon as a service permission to the user. After I added the required permissions to the user I reinstalled everything on both nodes. When I reinstalled SQL Server I got the same authentication error. Insert annoyed emoji here.

This is obviously some sort of permissions problem. So I add domain admin and enterprise admin privileges to the service account. Once again I reinstall everything on the two failover nodes. This time I install SQL Server using the uber-permissions service account and I still get the same authentication error.

So I do some searches on Google regarding SQL Server 2014 installation and Windows Server 2012. This authentication error is a known issue with Windows Server 2012 and SQL Server 2014. The recommendation to get the software installed is to use the **NT AUTHORITY \NetworkService** account then change the account in services later. So once again I reinstall everything. This time during the SQL Server installation, on the screen where I enter the service accounts I use **NT AUTHORITY \NetworkService**. When I click on the next button I get the error message saying that the service accounts must be a domain account. The reason I got this error and the people who recommended this fix did not is because I'm installing onto a failover cluster not just onto a Windows Server.

After lots and lots of searching the internet I find someone who had the same problem on a similar setup. He tested installing a failover cluster on Windows Server 2012 with SQL Server 2014 & 2008. Both SQL Server versions failed to install with authentication errors. Then he tried installing a failover cluster on Windows Server 2008 with SQL Server 2014 & 2008. Both SQL Server versions installed perfectly. One other item he mentioned on the blog post was VirtualBox snapshots.

On the Oracle VM VirtualBox Manager screen there is a little camera icon labeled Snapshots. I thought snapshots would be like a screen shot since it has the camera icon. Actually a snapshot saves the state of the virtual PC. Any changes to the state are saved in a separate file. This means you can get rid of the current state and go back to a previous state. Practically for me this means that I could have taken a snapshot of the domain controller and failover nodes before I installed SQL Server. That way if the installation failed I could return to the state before the installation and not need to reinstall everything.

From DreamSpark I download the Windows Server 2008 R2 software. Then I reset the permissions for the system account to be just a domain user and have logon as a service permission. Next I remove both failover nodes from the domain. Finally I install Windows Server 2008 R2 on both PCs. I add both machines to the domain. Next I setup the failover cluster services on both nodes. Then I install all of the updates from Microsoft. After the updates I take snapshots of both nodes and the domain controller. Finally I install SQL Server on the cluster. The installation works perfectly. I now have a SQL Server failover cluster.

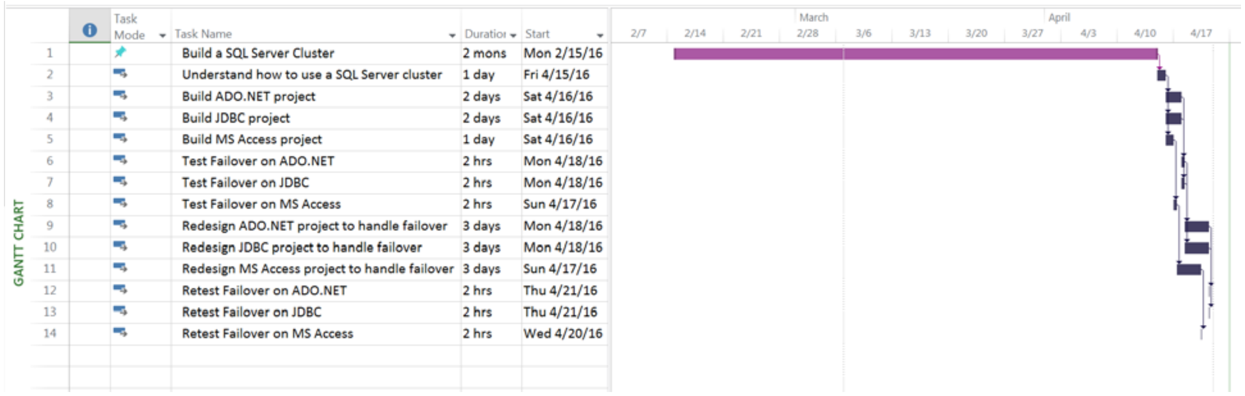
Client Computer

For the client computer I installed Windows 8.1 for the operating system. Also the client computer had 4 GB ram, 4 processors, a network card, and an 80 GB hard drive. On the client computer I installed the following software.

- Visual Studio 2015
- Eclipse
- Java Development Kit
- Microsoft Access 2016
- SQL Server Management Tools

2.3 Timeline

Gantt Chart



		Task Mode	Task Name	Duration	Start
1			Build a SQL Server Cluster	2 mons	Mon 2/15/16
2			Understand how to use a SQL Server cluster	1 day	Fri 4/15/16
3			Build ADO.NET project	2 days	Sat 4/16/16
4			Build JDBC project	2 days	Sat 4/16/16
5			Build MS Access project	1 day	Sat 4/16/16
6			Test Failover on ADO.NET	2 hrs	Mon 4/18/16
7			Test Failover on JDBC	2 hrs	Mon 4/18/16
8			Test Failover on MS Access	2 hrs	Sun 4/17/16
9			Redesign ADO.NET project to handle failover	3 days	Mon 4/18/16
10			Redesign JDBC project to handle failover	3 days	Mon 4/18/16
11			Redesign MS Access project to handle failover	3 days	Sun 4/17/16
12			Retest Failover on ADO.NET	2 hrs	Thu 4/21/16
13			Retest Failover on JDBC	2 hrs	Thu 4/21/16
14			Retest Failover on MS Access	2 hrs	Wed 4/20/16

Figure 2-7 Timeline

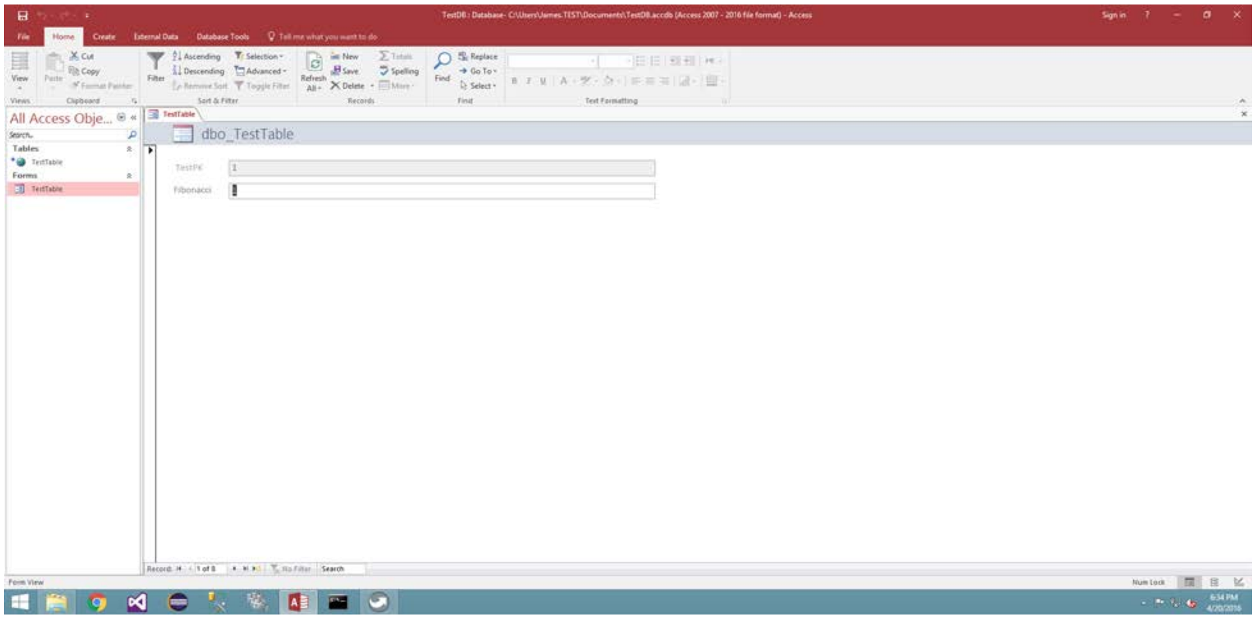
To build the clustered server I expected it would take two days unfortunately due to the issues I ran into it took two months. Then another one day to make sure the clustered server works as expected. Next I expect that it will take me three days to create a MS Access project, a Java project, and a C# project. After that it will take me a day to test the three projects with a SQL Server failover. Then I'm estimating another three days to modify the three projects to handle the failover. Finally I'll have one more day of testing the projects on a failover.

Chapter 3

3.1 User Interface

Microsoft Access

In Microsoft Access I created a single form.


TestDB : Database- C:\Users\James.TEST\Documents\TestDB.accdb (Access 2007 - 2016 file format) - Access

dbo_TestTable

TestPK	1
Fibonacci	

Figure 3-1 Access Form

The form connected to the TestTable table linked from the SQL Server failover cluster. The TestTable had two columns. The first field is the primary key and is assigned a value automatically. The second field is an integer field and gets its value from the text box on the form. For fun I decided to just put Fibonacci numbers there.

Java

The Java application is just a console application with no real user interface. The program would connect to the TestTable on the SQL Server failover cluster. Then interate through the records printing the records to the console. Next calculate the value of the next Fibonacci number and add that number to the table.

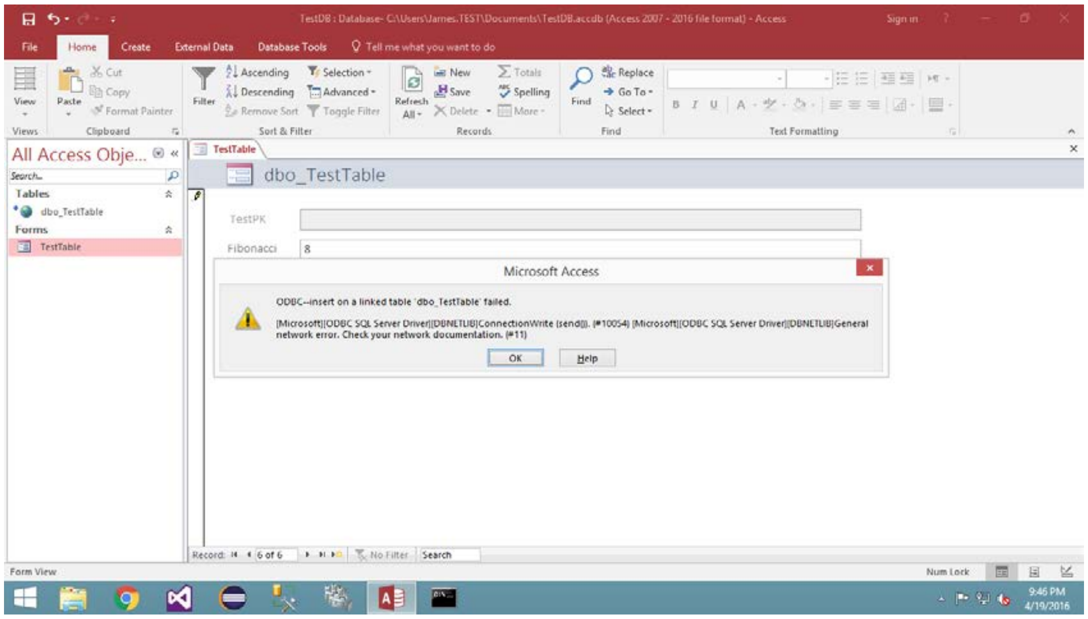
C#

The C# application is also a console application with no real user interface. The program connected to the TestTable on the SQL Server failover cluster. Same as the Java program this program would iterate through all the records printing them to the console. Next calculate the value of the next number and add the new number back to the table.

3.2 Testing Methodology

Microsoft Access

At various points of using the form to add, update and delete records I would stop what I was doing and cause a failover on the SQL Server.



Fibonacci: 8

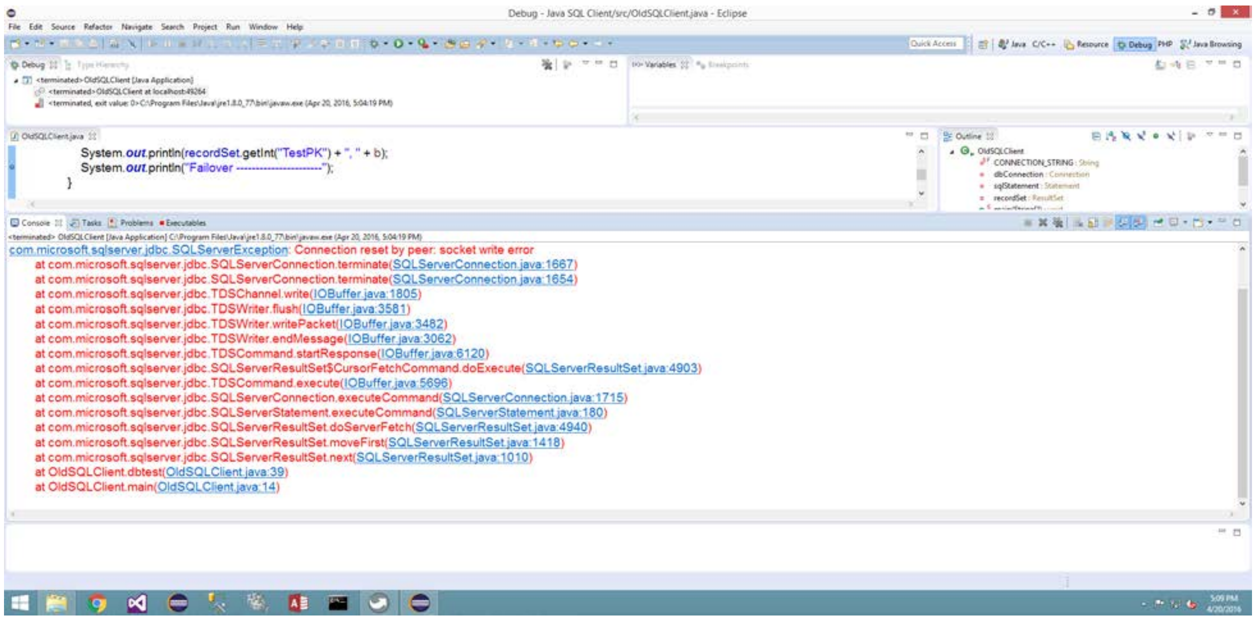
```
ODBC--insert on a linked table 'dbo_TestTable' failed.

[Microsoft][ODBC SQL Server Driver][DBNETLIB]ConnectionWrite (send()). (#10054) [Microsoft][ODBC SQL Server Driver][DBNETLIB]General network error. Check your network documentation. (#11)
```

Figure 3-2 Access Error Message

If I opened the form and made no changes on the form data and caused a failover event then after a minute I would get an ODBC error with no message. Other than that I would get an error like the above error.

Java

At various places the Java program would print to the console “Failover ----------------------“. At these points there would be a debug break point set. When I hit that break point I would cause a failover of the SQL Server. Then continue the program. Then I would get an error like the following.



```
<terminated>OldSQLClient [Java Application]
    <terminated>OldSQLClient at localhost:49264
    <terminated, exit value: 0>C:\Program Files\Java\jre1.8.0_77\bin\javaw.exe (Apr 20, 2016, 5:04:19 PM)
```

```
            System.out.println(recordSet.getInt("TestPK") + ", " + b);
            System.out.println("Failover ----------------------");
        }
```

```
<terminated> OldSQLClient [Java Application] C:\Program Files\Java\jre1.8.0_77\bin\javaw.exe (Apr 20, 2016, 5:04:19 PM)
com.microsoft.sqlserver.jdbc.SQLServerException: Connection reset by peer: socket write error
	at com.microsoft.sqlserver.jdbc.SQLServerConnection.terminate(SQLServerConnection.java:1667)
	at com.microsoft.sqlserver.jdbc.SQLServerConnection.terminate(SQLServerConnection.java:1654)
	at com.microsoft.sqlserver.jdbc.TDSChannel.write(IOBuffer.java:1805)
	at com.microsoft.sqlserver.jdbc.TDSWriter.flush(IOBuffer.java:3581)
	at com.microsoft.sqlserver.jdbc.TDSWriter.writePacket(IOBuffer.java:3482)
	at com.microsoft.sqlserver.jdbc.TDSWriter.endMessage(IOBuffer.java:3062)
	at com.microsoft.sqlserver.jdbc.TDSCommand.startResponse(IOBuffer.java:6120)
	at com.microsoft.sqlserver.jdbc.SQLServerResultSet$CursorFetchCommand.doExecute(SQLServerResultSet.java:4903)
	at com.microsoft.sqlserver.jdbc.TDSCommand.execute(IOBuffer.java:5696)
	at com.microsoft.sqlserver.jdbc.SQLServerConnection.executeCommand(SQLServerConnection.java:1715)
	at com.microsoft.sqlserver.jdbc.SQLServerStatement.executeCommand(SQLServerStatement.java:180)
	at com.microsoft.sqlserver.jdbc.SQLServerResultSet.doServerFetch(SQLServerResultSet.java:4940)
	at com.microsoft.sqlserver.jdbc.SQLServerResultSet.moveFirst(SQLServerResultSet.java:1418)
	at com.microsoft.sqlserver.jdbc.SQLServerResultSet.next(SQLServerResultSet.java:1010)
	at OldSQLClient.dbtest(OldSQLClient.java:39)
	at OldSQLClient.main(OldSQLClient.java:14)
```

Figure 3-3 Java Error Message

C#

For the C# application I did not use debug break points. For the locations in the program I wanted to test a failover event I would call a function that would print to the console “Failover ----------------------“ and wait for the user to press the enter key. Before I hit the enter key I would cause a failover event on the cluster.

Chapter 4

4.1 What’s this all about?

This project has 4 parts. The first part is building a SQL Server Failover Cluster. The Second part is testing the cluster to make sure that the cluster is setup properly. The third part of this project is testing to see exactly what happens to applications that are connected to SQL when a failover event happens. The final part of this project was for me to get a SQL Server certification from Microsoft. I expected to get the first three parts done in 3 or 4 weeks. Then I planned to spend the rest of the semester studying for the certification.

4.2 Building the Cluster

The first thing I needed to do was get the software I would need to install on the severs. Since I was a student in college I could download most of the software I would need from Microsoft without using the UAA Dreamspark account. So I downloaded Windows Server 2012 and SQL Server 2014.

Once I got the software, I setup 3 virtual PCs. The first PC I installed OpenFiler. The next two PCs I installed Windows Server 2012. Then I configured one PC to be a domain controller. Next I tried to add the failover cluster services to the two nodes. During the tests that get run I got an error saying that one node had active directory services and the other node did not. So I could not install the failover cluster services.

I added a fourth PC and installed Windows Server 2012 R2 on it. Then I tried to install the failover cluster services on this PC and the other Windows Server 2012 PC. Again I got an error, both machines must be running the exact same version of Windows Server. Time to start over.

I formatted the three Windows PCs. The first PC installed Windows Server 2012 and called it DC. Then I added active directory services to the DC computer.

The second and third PCs I installed Windows Server 2012 R2 and called them SQL1 and SQL2 respectively. These would be the failover nodes. After a lot of attempts to get the SQL failover cluster installed I ended up installing Windows Server 2008 R2 on these PCs. Also I replaced OpenFiler with FreeNAS.

Next I needed at client PC. I could not download Windows 8.1 from Dreamspark with my current access so I contacted Dr. Mock about getting me access through UAA's Dreamspark account. Once I had that access I downloaded Windows 8.1, Microsoft Access 2016 and Visual Studio 2015. Then I setup a fifth virtual PC as the client and installed Windows and the software I needed to run my failover tests.

4.3 Testing the Cluster

After all the excitement of building the cluster, running the tests was a simple task. On the client PC I ran the SQL Server Management Studio software and connected it to the SQL Server. Then on the cluster I caused a failover. After that the management studio took a minute to reconnect but it worked fine. Then I added 3 databases each with 1 table to SQL Server. Next I caused another failover event. The three databases were still there after the failover so everything is working fine.

4.4 Testing Failover on Applications

Microsoft Access

The first application that I created and tested was a Microsoft Access database that linked to the table on the SQL Server. In Access I created a form to add, update, and delete records in the linked table.

To run the tests I first opened the form then would start manipulating the records. At an appropriate stage where the record would need to be sent to the server I would cause a failover event. From this I could see the error message that got generated.

Java

The second application I created was a Java console application. With this application I used the debug tools available in Eclipse to help me do the tests. The application would connect to the SQL table and retreive a result set. Then it would iterate through the result set and print the records to the console. Finally it calculated the next Fibonacci number and added it as a record back to the table. At various statements I put break points and caused a failover event at those points. All of the errors were basically the same, the network socket got reset.

C#

The final application I created was a C# console application. For this application I did not use the debug tools available in Visual Studio. Instead I had the application print to the console window at the locations I wanted to cause a failover then wait for a carrage return. At each of these prompts I would cause a failover event.

Chapter 5

5.1 Summary

One of the tasks for this project was for me to get a certification in SQL Server. Unfortunately this task was not completed. Building the cluster took far longer than I expected so the amount of time available to study for the certification was not enough. That's the bad news. The good news is that the rest of the project went very well.

First I was able to successfully build and test a SQL Server failover cluster. Once I had learned the hard way all of the lessons regarding setting up the cluster. I was able to setup everything from scratch in about 2 days.

Finally testing the Access, Java and C# applications went just fine.

Microsoft Access

With my Access application I was unable to get it to reconnect to the linked table programmatically. The only way that worked was to exit out of the application and start it up again.

Java

The Java application gave me errors whenever the application tried to interact with the cluster after a failover event. Each time I was able to recover the connection programmatically.

C#

Of the three applications I created the C# application gave me the most suprising results. I tested this application with failover events in the same relative spots as the Java application. The difference is I received no errors and the connection stayed up. For all of the tests I could not break the connection. So kudos to Microsoft.

I have a hypothesys as to why the C# connection never went down. SQL Azure is a cloud based version of SQL Server. Since Azure is cloud based the connection could be a little flaky since it goes through the internet. Therefore I think Microsoft spent some extra effort on making sure the ADO.NET connections would stay connected.

5.2 Implications

First implication, don’t use Microsoft Access for high availability applications. Second if you are going to use Java for a high availability application then you need to spend some effort in making sure the connection is available at all times. Finally for high availability applications C# would be the best choice.

5.3 Recommendations for future testing

For this project I tested only a failover cluster but there are many other ways to setup SQL for high availability. First there is a newer technology called AlwaysOn. This should be tested in the same manner as my tests. Second there is SQL Azure the cloud based server that could be tested.

Also I only tested with Microsoft SQL Server. These tests could be done for MySQL, Oracle, Amazon Relational Database Service, and many other databases.

Bibliography

Gosling, J., Joy, B., Steele, G., Bracha, G., & Buckley, A. (2015). *The Java® Language Specification* (Java SE 8 Edition ed.). Redwood City, California: Oracle America, Inc. Retrieved from https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf

Gregory Kincade. (2015, 2 11). *Matching data requirements with flexible storage*. Retrieved 2 1, 2016, from Dell TechCenter: http://en.community.dell.com/techcenter/b/techcenter/archive/2015/02/11/matching-data-requirements-with-flexible-storage

Identiv. (2015). *uTrust Velocity Software*. Retrieved 2 3, 2016, from Identive: http://www.identiv.com/products/premises/utrust-software

Microsoft. (2012, 6 11). *Administering Microsoft SQL Server 2012 Databases*. Retrieved 2 3, 2016, from Microsoft Learning: https://www.microsoft.com/en-us/learning/exam-70-462.aspx

Microsoft. (2014). Introducing Microsoft SQL Server 2014 Technical Overview. In R. Mistry, & S. Misner, *Introducing Microsoft SQL Server 2014 Technical Overview* (p. xii). Redmond: Microsoft Press.

Microsoft. (2016). *Introduction to the C# Language and the .NET Framework*. Retrieved 4 23, 2016, from msdn.microsoft.com: https://msdn.microsoft.com/en-us/library/z1zx9t92.aspx

Microsoft. (2016). *Technet Library - Windows Server*. Retrieved 4 23, 2016, from technet.microsoft.com: https://technet.microsoft.com/en-us/library/bb625087.aspx

Openfiler. (2015). *Products*. Retrieved 2 3, 2016, from Openfiler: https://www.openfiler.com/products

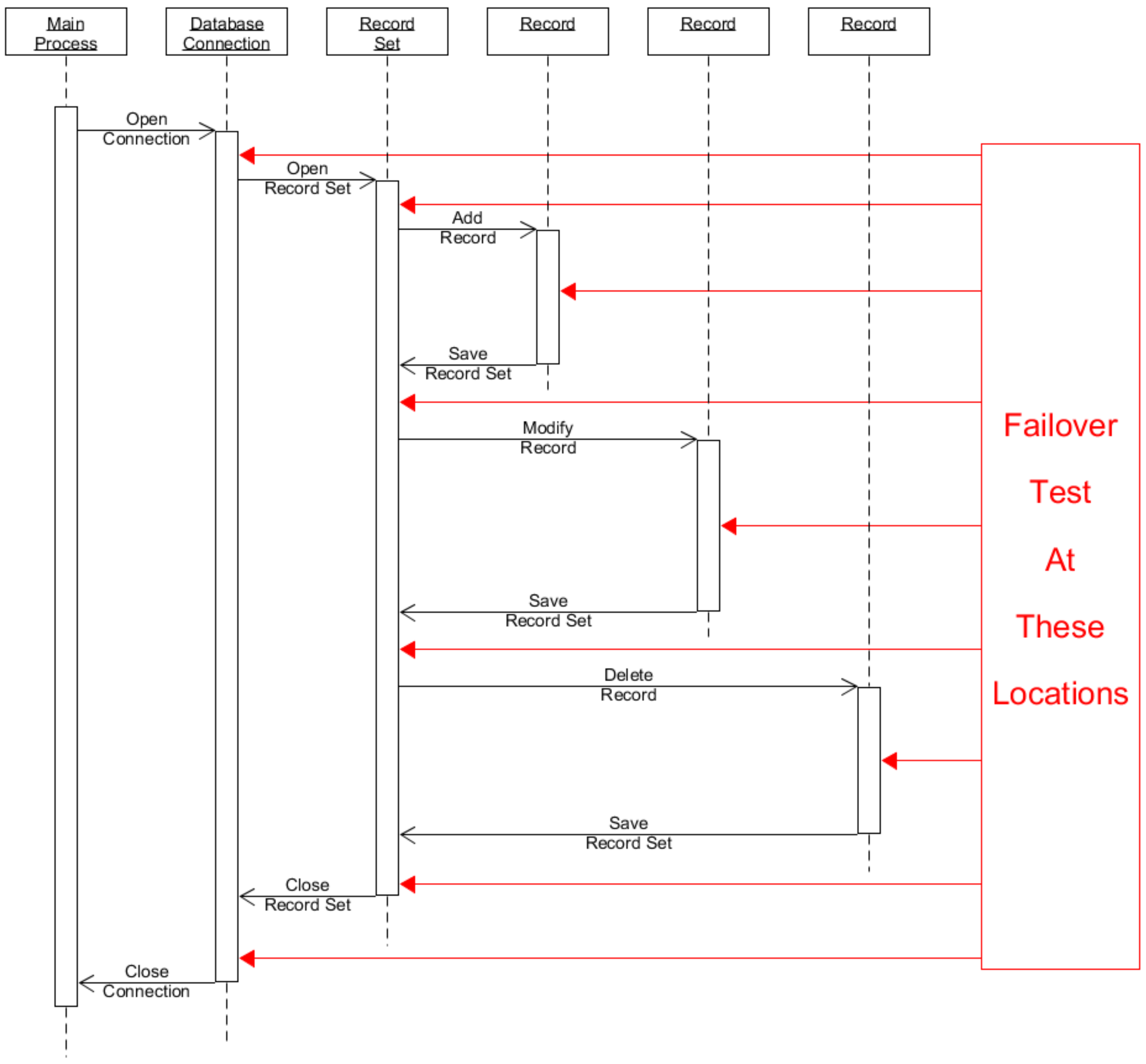
Oracle Corporation. (2016). *manual/ch01.html*. Retrieved 4 23, 2016, from www.virtualbox.org: https://www.virtualbox.org/manual/ch01.html

PC.net. (2016). *Access*. Retrieved 4 23, 2016, from PC.net Glossary: http://pc.net/glossary/definition/access

Wikipedia. (2015, 12 12). *Microsoft Cluster Server*. Retrieved 4 23, 2016, from en.wikipedia.org: https://en.wikipedia.org/wiki/Microsoft_Cluster_Server

Wikipedia. (2016, 1 29). *Access control*. Retrieved 2 1, 2016, from Wikipedia: https://en.wikipedia.org/wiki/Access_control

Appendix A UML Diagram



Appendix B Source Code

Github address

https://github.com/JamesTweet/Capstone-2016

Microsoft Access Code

```
Option Compare Database
Private Sub fixLinks()
        Dim tdf As TableDef
        For Each tdf In CurrentDb.TableDefs
                If tdf.Connect <> vbNullString Then
                tdf.Connect = tdf.Connect & ""
                tdf.RefreshLink
                'tdf.Connect = "DRIVER=SQL Server;Database =  _ MS_Access_Database;APP=Microsoft
                Office 2016;Trusted_Connection = _ Yes;SERVER = SQL"
                'tdf.RefreshLink
                End If
        Next
        CurrentDb.TableDefs.Refresh
End Sub

Private Sub Form_BeforeQuery()
        fixLinks
End Sub

Private Sub Form_BeforeUpdate(Cancel As Integer)
        fixLinks
End Sub

Private Sub Form_Load()
        fixLinks
End Sub

Private Sub Form_OnConnect()
        fixLinks
End Sub

Private Sub Form_Open(Cancel As Integer)
        fixLinks
End Sub

Private Sub Form_Query()
        fixLinks
End Sub
```

Java Code

```
import java.sql.*;
public class OldSQLClient {
  private static final String CONNECTION_STRING =
  "jdbc:sqlserver://SQL;DatabaseName=Java_Database;integratedSecurity=true;";
  private Connection dbConnection;
  private Statement sqlStatement;
  private ResultSet recordSet;
  public static void main(String[] args) {
      try {
          Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
          OldSQLClient s = new OldSQLClient();
          s.dbtest();
      } catch (ClassNotFoundException e) {
          e.printStackTrace();
      }
  }
  private void dbtest() {
      int a = 0;
      int b = 0;
      try {
          dbConnection = DriverManager.getConnection(CONNECTION_STRING);
          System.out.println("Failover ----------------------");
          if (dbConnection.isClosed() == true || dbConnection.isValid(60) == false )
              dbConnection = DriverManager.getConnection(CONNECTION_STRING);

          String query = "SELECT * FROM dbo.TestTable";
          sqlStatement = dbConnection.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
          System.out.println("Failover ----------------------");
          if (dbConnection.isClosed() == true || dbConnection.isValid(60) == false ){
              dbConnection = DriverManager.getConnection(CONNECTION_STRING);
              sqlStatement = dbConnection.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
          }
          recordSet = sqlStatement.executeQuery(query);
          System.out.println("Failover ----------------------");
          while (recordSet.next()) {
              a = b;
              b = recordSet.getInt("Fibonacci");
              System.out.println(recordSet.getInt("TestPK") + ", " + b);
              System.out.println("Failover ----------------------");
          }

          System.out.println("Failover ----------------------");
          recordSet.moveToInsertRow();
          System.out.println("Failover ----------------------");
          recordSet.updateInt("Fibonacci", a+b);
          System.out.println("Failover ----------------------");
          recordSet.insertRow();

          System.out.println("Failover ----------------------");
          recordSet.close();
```

```
            sqlStatement.close();
            dbConnection.close();

        } catch (SQLException e) {
            e.printStackTrace();
            System.exit(0);
        }


    }
}
```

C# Code

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;

namespace SQL_Failover_Test
{
   class Program //:: StateChangeEventHandler
   {
      private SqlConnection dbConnection;
      private int failCount = 0;

      static void Main(string[] args)
      {
         Program p = new Program();
         p.Test();

      }

      private void Test()
      {
         int a = 0;
         int b = 0;

         dbConnection = new SqlConnection(
         "Server=SQL;Database=C_Sharp_Database;Trusted_Connection=true");
         dbConnection.InfoMessage += new SqlInfoMessageEventHandler(OnInfoMessage);
         dbConnection.Open();
         Failover("Connection Open");

         SqlCommand command = new SqlCommand("SELECT * FROM dbo.TestTable", dbConnection);

         Failover("SQL Command Created");
         using (SqlDataReader reader = command.ExecuteReader())
         {
            Console.WriteLine("PK\tFibonacci");
            while (reader.Read())
            {
               b = a;
               a= (int) reader[1];
               Console.WriteLine(String.Format("{0} \t | {1}",
                  reader[0], a));
               //Failover("Reading Records");
            }
         }

         SqlCommand insertCommand = new SqlCommand("INSERT INTO dbo.TestTable (Fibonacci) VALUES
         (@0)", dbConnection);
         insertCommand.Parameters.Add(new SqlParameter("0", a + b));
         Failover("Execute Insert");
         insertCommand.ExecuteNonQuery();
```

```
            dbConnection.Close();

            Failover("Close");

        }

        private void Failover(String location)
        {
            failCount++;
            Console.WriteLine(location + " Failover ---------------------- " + failCount.ToString());
            Console.ReadLine();
        }

        protected static void OnInfoMessage(object sender, SqlInfoMessageEventArgs args)
        {
            Console.WriteLine("SQL Info Message");

            foreach (SqlError err in args.Errors)
            {
                Console.WriteLine("The {0} has received a severity {1}, state {2} error number {3}\n" +
                                  "on line {4} of procedure {5} on server {6}:\n{7}",
                                  err.Source, err.Class, err.State, err.Number, err.LineNumber,
                                  err.Procedure, err.Server, err.Message);
            }
        }

    }
}
```

Source Code License

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or