Information Management software

Automating DB2 HADR Failover on Windows using Tivoli System Automation for Multiplatforms

December 2012

Authors:

Steve Raspudic, IBM Toronto Lab (stevera@ca.ibm.com)
Michelle Chiu, IBM Toronto Lab (mtmchiu@ca.ibm.com)
Philippe Stedman, IBM Toronto Lab (pstedman@ca.ibm.com)

Table of Contents

1. Introduction and Overview..... 3

2. Before You Begin..... 3
2.1 Knowledge Requirements..... 3
2.2 Software Configuration Used..... 3

3. Overview of Important Concepts..... 4
3.1 HADR Overview..... 4
3.2 SA MP Overview..... 4
3.3 Typical HADR Topology..... 4

4. Steps to Set Up Topology..... 6
4.1 Basic Network Setup..... 6
4.2 Install DB2 Software..... 8
4.3 Install SA MP..... 9
4.4 Prepare SA MP Cluster..... 9
4.5 Create a DB2 HADR Database..... 10
4.6 Register HADR with SA MP for Automatic Management..... 13

5. Post Configuration Testing..... 15

6. Testing Topology Response to Common Failures..... 17
6.1 Controlled Failover Testing..... 17
6.2 Testing Instance Failure: Primary Instance..... 17
6.3 Testing Instance Failure: Standby Instance..... 18
6.4 Testing Resource Group Failure: Primary Instance Resource Group 19
6.5 Testing Resource Group Failure: Standby Instance Resource Group 20
6.6 Testing Network Adapter Failure..... 21
6.7 Node Failure..... 22

7. Conclusion..... 27

Appendix A: Understanding How SA MP Works..... 28
Appendix B: Troubleshooting Tips..... 30
Appendix C: Resources and Resource Groups Setup Scripts..... 32
Appendix D: Automated HADR Reintegration..... 50

1. Introduction and Overview

This paper will guide you through the implementation of an automated failover solution for the IBM® DB2® Enterprise Server Edition for Linux, UNIX, and Windows Version 9.5 database server (DB2 9.5) product. The solution is based on a combination of the high availability disaster recovery (HADR) feature in DB2 9.5 and the IBM Tivoli® System Automation for Multiplatforms product (SA MP). The setup described in this paper focuses on the Windows operating system.

Target audience for this white paper:
- DB2 database administrators
- Windows system administrators

2. Before You Begin

Below you will find information on knowledge requirements, as well as hardware and software configurations used to set up the topology depicted in this paper. It is important that you read this section prior to beginning any setup.

2.1 Knowledge Requirements

- Basic knowledge of DB2 9.5 software and the HADR* feature.
- Basic understanding of SA MP cluster manager software**
- Basic understanding of Windows administration concepts

*Information on the DB2 HADR feature can be found here:
http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp

**Information on SA MP can be found here:
http://www.ibm.com/software/tivoli/products/sys-auto-multi/

2.2 Software Configuration Used

Windows Server 2003 Release 2 Enterprise Edition was used in this configuration.
For information about software requirements for running SA MP, refer to:
http://www.ibm.com/software/tivoli/products/sys-auto-linux/platforms.html

Listed below are the components of the software configuration used to set up the environment for this paper:
- Operating system: Windows Server 2003 Release 2 Enterprise Edition
- DB2 product: IBM DB2 Enterprise Server Edition (ESE) Version 9.5 Fix Pack 7
- Tivoli product: IBM Tivoli System Automation for Multiplatforms Version 3.1 with Fix Pack 7

3. Overview of Important Concepts

3.1 HADR Overview

The HADR feature of DB2 9.5 allows a database administrator (DBA) to have one "hot standby" copy of any DB2 database such that, in the event of a primary database failure, a DBA can quickly switch over to the "hot standby" with minimal interruption to database clients. (See Fig.1 below for a typical HADR environment.)

However, an HADR primary database does not automatically switch over to its standby database in the event of failure. Instead, a DBA must manually perform a takeover operation when the primary database has failed.

3.2 SA MP Overview

Since an HADR primary database does not automatically switch over to its standby database in the event of failure, to achieve automatic monitoring and failover, a DBA must set up SA MP with DB2. For example, the topology shown in Fig. 1 below would perform automatic failover.

During the SA MP configuration process, the necessary HADR resources and their relationships are defined to the cluster manager. Failure events in the HADR system can then be detected automatically, and takeover operations can be executed without manual intervention.

3.3 Typical HADR topology

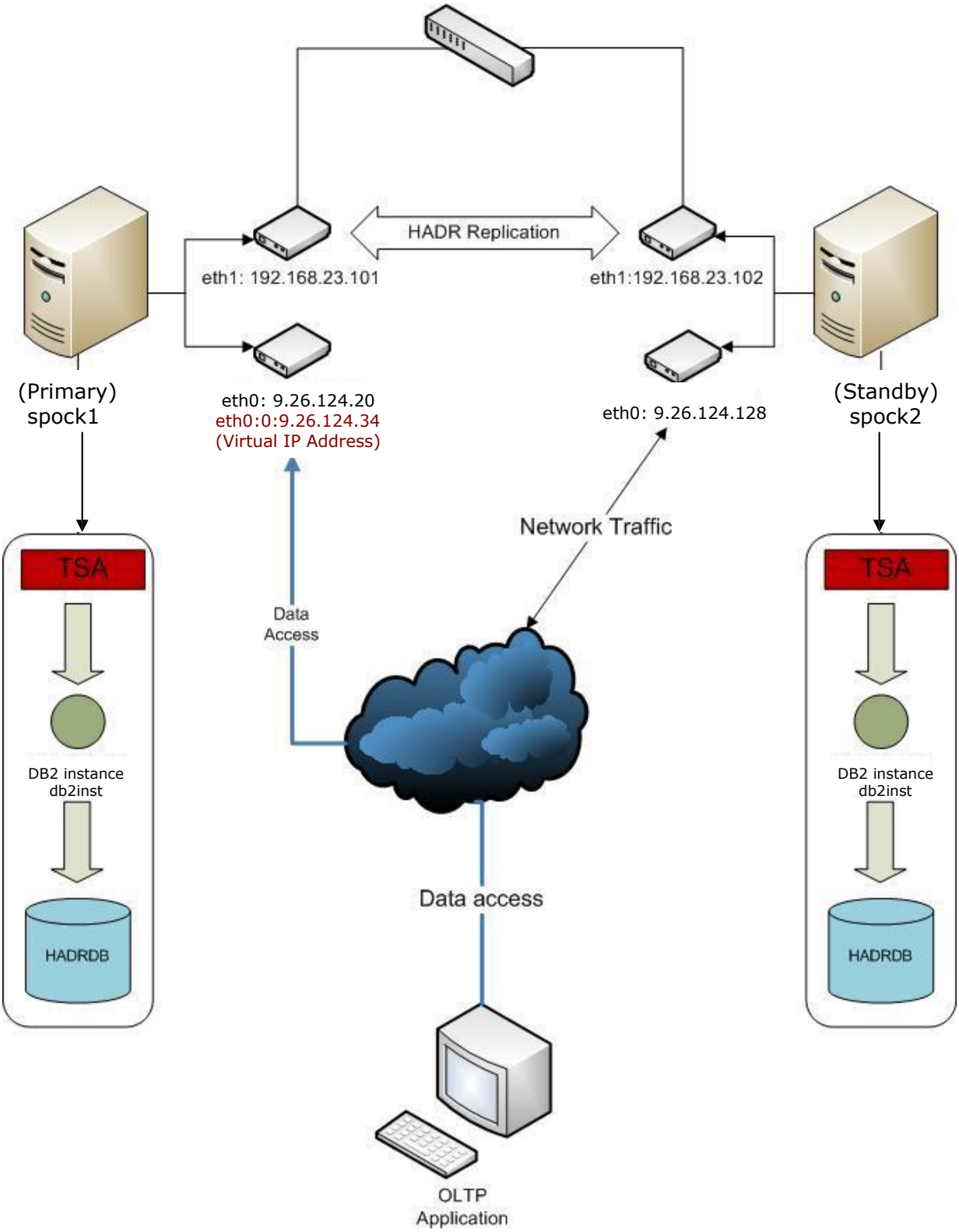
A typical HADR topology contains two nodes: a primary node to host the primary HADR database, and a standby node to host the standby HADR database. The nodes are connected to each other over a network to accommodate transaction replication between the two databases.

A single network HADR topology is implemented in this paper.

In Fig. 1, SA MP now monitors the HADR pair for primary database failure, and will issue appropriate takeover commands on the standby database in the event of a primary database failure.

To gain a better understanding of how SA MP works, read Appendix A at the end of this paper.

Fig. 1. Typical HADR Environment with SA MP



4. Steps to Set Up Topology

The following section documents a two-node topology, in which one node (e.g., spock1) hosts the primary database (e.g., HADRDB), and a second node (e.g., spock2) hosts its standby. Complete the following steps to set up the topology depicted in Fig. 1 above.

Notes:

1. Your topology does *not* have to include redundant network interface cards (NICs) (e.g., eth1 in Fig. 1 above). Redundant NICs allow for recovery from simple outages caused by primary NIC failure (e.g., eth0). For example, if eth0 on spock1 in Fig. 1 failed for some reason, then the IP address that it was hosting (i.e., 9.26.124.20) could be taken over by eth1. In fact, there is an opportunity in Section 4.4 step 7 to make the IP address of each DB2 instance (e.g., db2inst) highly available with SA MP.

2. The letters in front of a command in the following steps designate on which node a command is to be issued to properly set up the topology shown in Fig. 1 above. The order of the letters also designates the order in which you should issue a command on each node:

   (P) = Primary Database Node (e.g., spock1)

   (S) = Standby Database Node (e.g., spock2)

3. The parameters given for commands in this paper are based on the topology shown in Fig. 1 above. Change the parameters accordingly to match your specific environment. Also, a "\" in a command designates that the command text continues on the next line (i.e., *do not* include the "\" when you issue the command).

4.1 Basic Network Setup

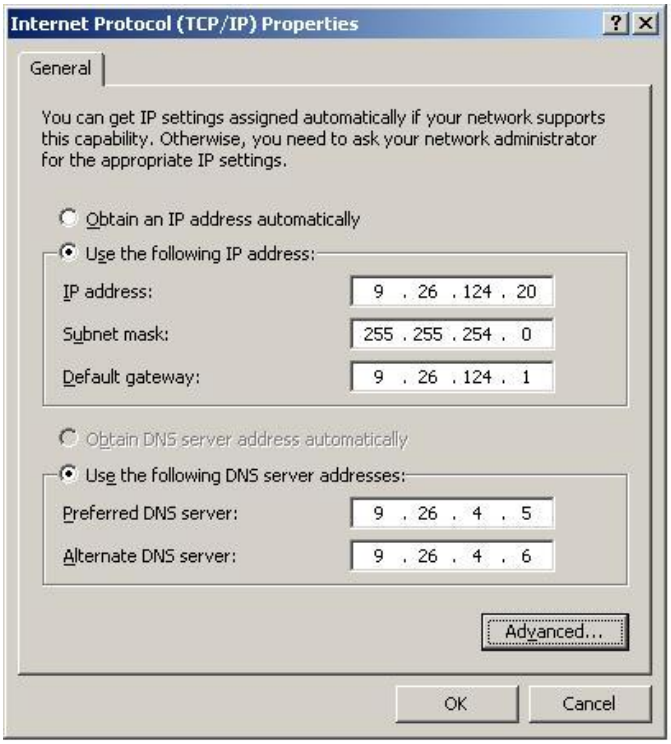
Make sure that all nodes (e.g., spock1, spock2) will be able to communicate with each other via TCP/IP protocol.

1. Set up the network using a static IP address. (We use a static IP address in Fig. 2 with a subnet mask of 255.255.254.0 for each node.)

   Start > Control Panel > Network Connections

   Right-click **Local Network Connection** and select **Properties**. In the "This connection uses the following items" box, highlight "Internet Protocol," and select **Properties**.

Fig. 2: Internet Protocol (TCP/IP) Properties



In Fig. 2, click **Advanced** to enter the Advanced TCP/IP Setting, on the WINS tab add the WINS address (see Fig. 3). Also on the DNS tab click **Append these DNS suffixes** and then click **Add** (see Fig. 4).

Note: IP address, DNS address, WINS address, and DNS suffixes may be different for your system.

Fig. 3: Advanced TCP/IP Setting (WINS)

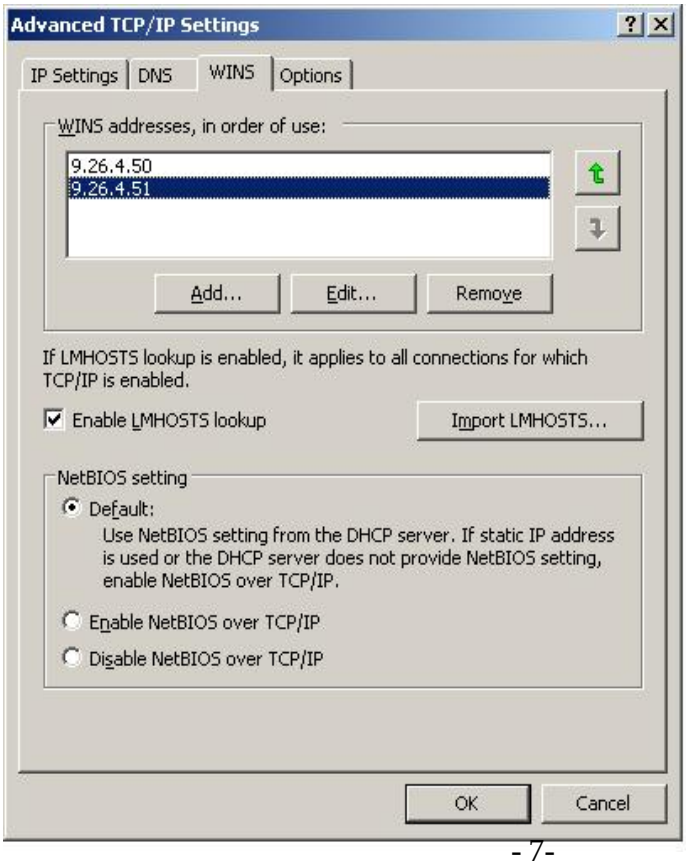
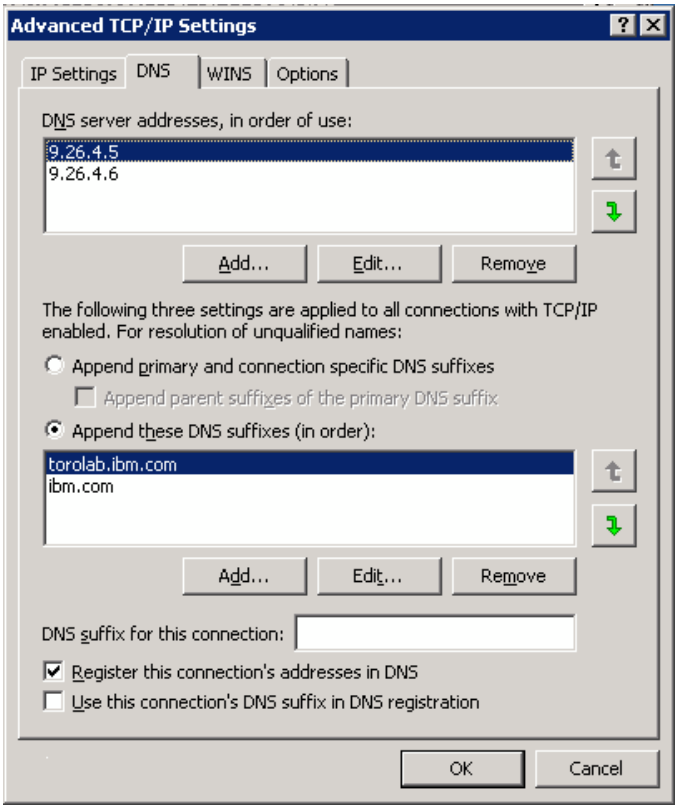


Fig. 4: Advanced TCP/IP Setting (DNS)



2. Turn off the Firewall on each node to make HADR pair to connect each other:

   Start > Control Panel > Windows Firewall. Click **Off** and then **OK**.

3. Test that you can ping from each node to all other nodes successfully using the following commands:

```
(P)(S) $ ping spock1
(P)(S) $ ping spock2
(P)(S) $ ping spock1.torolab.ibm.com
(P)(S) $ ping spock2.torolab.ibm.com
(P)(S) $ ping 9.26.124.20
(P)(S) $ ping 9.26.124.128
```

4.2 Install DB2 Software

As the user name user1, install DB2 ESE Version 9.5 Fix Pack 7 software on the primary and standby nodes (e.g., spock1 and spock2).

Follow the link to install the DB2 software on a Windows platform:
http://www.ibm.com/developerworks/db2/library/techarticle/0302gao/0302gao.html

Important: The DB2 SA MP scripts for HADR on Windows require that the *same* instance name be used for the primary and standby instances.

4.3 Install SA MP

Install SA MP V3.1, and then upgrade to Fix Pack 7 (i.e., SA MP V3.1.0.7) on the nodes, by following these steps:

1. Follow "Chapter 2: Installing System Automation for Multiplatforms on Windows" of the SA MP-Inst-Config-Guide.pdf (SAM3107MPWindows/docs).

2. As user1, the local user, go to the directory where the SA MP 3.1 installation .exe file and SA MP license exist, and then run the SA MP 3.1 installer.

3. Use the "IBM Tivoli System Automation – Shell" to add the appropriate IP address to the host name mappings in the /etc/hosts file of each node (P), and (S):

   SA MPle content of /etc/hosts on (P), and (S):

   ```
   9.26.124.20 spock1.torolab.ibm.com spock1
   9.26.124.128 spock2.torolab.ibm.com spock2
   ```

   Adding static IP address to host name mappings to the hosts file removes the systems DNS servers as a single point of failure. If DNS fails, the cluster systems can still resolve the addresses of the other machines via the hosts file.

4.4 Prepare SA MP Cluster

Make sure that all SA MP installations in your topology know about one another, and can communicate with one another in what is referred to as a SA MP cluster domain. This is essential for management of HADR by SA MP.

1. Using the "IBM Tivoli System Automation – Shell," run the following command as local user to prepare the proper security environment between the SA MP nodes:

   ```
   (P)(S) $ preprpnode spock1 spock2
   ```

2. Issue the following command to create the cluster domain:

   ```
   (P) $ mkrpdomain hadr_domain spock1

   spock2 (P) $ lsrpdomain
         Name        OpState RSCTActiveVersion MixedVersions TSPort
         GSPort hadr_domain Offline 2.5.5.2    No            12347  12348
   ```

3. Now start the cluster domain as follows. (**Note:** all future SA MP commands will be run relative to this active domain):

   ```
   (P) $ startrpdomain hadr_domain
   ```

4. Wait until hadr_domain is online by issuing the following command:

   ```
   (P)(S) $ lsrpdomain
         Name        OpState RSCTActiveVersion MixedVersions TSPort
         GSPort hadr_domain Online             2.5.5.2       No     12347
                     12348
   ```

5. Verify that all nodes are online in the domain as follows:

```
(P)(S) $ lsrpnode
         Name   OpState RSCTVersion
         spock1 Online  2.5.5.2
         spock2 Online  2.5.5.2
```

6. Go to the directory /usr/sbin/rsct/sapolicies/db2

```
(P)(S) $ ls
       db2.def    hadr_monitor.ksh    mkdb2
       db2ip.def  hadr_start.ksh      mkhadr
       hadr.def   hadr_stop.ksh       rmdb2
```

   If the files do not exist, follow the instructions in Appendix C. The definition files and mk scripts must be customized manually to suit your environment.

   Note: You will need the network equivalency only if you use a Service IP address to connect to the HADR database on the primary instance.

7. Create DB2 resources and equivalency:

```
(P) $ ./mkdb2

(P)(S) $ lssam

$ lssam
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
     '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
     '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
     |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
     '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

   Note: If the HADR database already exists and is in Peer mode, mkdb2 script will deactivate the database.

4.5 Create HADR Database

Now that you have created the primary and standby instances (db2inst), you need to create a database (for example, named HADRDB) that you will replicate with HADR.

All DB2 commands must be run from the db2cmd command prompt. To open the db2cmd window, select **Start > All Programs > IBM DB2 > DB2COPY > Command Line Tools > Command Window**.

Alternatively, run the **db2cmd** command at a Windows command prompt to open the db2cmd window.

For non-default instances, verify that the current instance is correct (i.e., db2inst in our example):

```
C:\Program Files\IBM\SQLLIB\BIN>echo %DB2INSTANCE%
DB2
C:\Program Files\IBM\SQLLIB\BIN>set DB2INSTANCE=db2inst

C:\Program Files\IBM\SQLLIB\BIN>echo %DB2INSTANCE%
db2inst
```

1. As user1, the local user, make sure that the database manager instance is started.

2. As primary instance owner (e.g., db2inst), create the database (e.g., HADRDB) that you will later make highly available with HADR as follows.

```
db2 create database hadrdb
```

3. Change the default Circular Logging of the database (e.g., HADRDB) to Archive Logging by issuing the following command:

```
db2 update db cfg for hadrdb using LOGRETAIN ON
```

4. Enable logging for index operations:

```
db2 update db cfg for hadrdb using LOGINDEXBUILD ON
```

5. As primary instance owner, enable HADR on the primary database as follows:

```
db2 update db cfg for hadrdb using HADR_LOCAL_HOST spock1
db2 update db cfg for hadrdb using HADR_REMOTE_HOST spock2
db2 update db cfg for hadrdb using HADR_LOCAL_SVC 55555
db2 update db cfg for hadrdb using HADR_REMOTE_SVC 55555
db2 update db cfg for hadrdb using HADR_REMOTE_INST db2inst
db2 update db cfg for hadrdb using HADR_SYNCMODE SYNC
db2 update db cfg for hadrdb using HADR_PEER_WINDOW 300
db2 update db cfg for hadrdb using HADR_TIMEOUT 120
```

6. Now we must create a backup copy of the primary database (e.g., HADRDB) that will later be restored on the standby instance, and act as the standby database of the HADR pair. The backup image will be written to the current directory (e.g., C:\Program Files\IBM\SQLLIB\BIN):

```
db2 backup database hadrdb
```

7. Transfer the backup image of the primary database (e.g., HADRDB) to the standby host machine.

8. As standby instance owner (e.g., db2inst), create the standby database on the standby node (e.g., spock2) as follows:

```
db2 restore database hadrdb replace history file
```

9. Allow TCP/IP communication to both the primary and standby instances (e.g., db2inst) as follows. **Important:** Make sure that "db2j_DB2 55000/tcp" is in the /etc/services (IBM Tivoli System Automation – Shell) file on the primary and standby nodes before issuing the following command:

```
db2set DB2COMM=tcpip
```

10. As standby instance owner (e.g., db2inst), enable HADR on the standby database (e.g., HADRDB) as follows:

```
db2 update db cfg for hadrdb using HADR_LOCAL_HOST spock2
db2 update db cfg for hadrdb using HADR_REMOTE_HOST spock1
db2 update db cfg for hadrdb using HADR_LOCAL_SVC 55555
db2 update db cfg for hadrdb using HADR_REMOTE_SVC 55555
db2 update db cfg for hadrdb using HADR_REMOTE_INST db2inst
```

11. As instance owner, verify that you have set the database configuration parameters correctly in steps 5 and 10 above:

```
db2 get db cfg for hadrdb | grep HADR

HADR database role                                      = STANDARD
HADR local host name                 (HADR_LOCAL_HOST)  = spock2
HADR local service name               (HADR_LOCAL_SVC)  = 55555
HADR remote host name                (HADR_REMOTE_HOST) = spock1
HADR remote service name              (HADR_REMOTE_SVC) = 55555
HADR instance name of remote server  (HADR_REMOTE_INST) = db2inst
HADR timeout value                       (HADR_TIMEOUT) = 120
HADR log write synchronization mode     (HADR_SYNCMODE) = SYNC
HADR peer window duration (seconds)  (HADR_PEER_WINDOW) = 300
```

12. As standby instance owner (e.g., db2inst), start HADR on the standby node (e.g., spock2) as follows:

```
db2 start hadr on db hadrdb as standby
```

As primary instance owner (e.g., db2inst), start HADR on the primary node (e.g., spock2) as follows:

```
db2 start hadr on db hadrdb as primary
```

13. As instance owner (e.g., db2inst), verify that the HADR pair is in Peer state as follows:

```
db2pd –hadr –db hadrdb

C:\Program Files\IBM\SQLLIB\BIN>db2pd -hadr -db hadrdb

Database Partition 0 -- Database HADRDB -- Active -- Up 0 days 00:02:32 -- Date
08/25/2010 10:07:46

HADR Information:
Role    State               SyncMode HeartBeatsMissed   LogGapRunAvg (bytes)
Standby Peer                Sync     0                  0
```

```
ConnectStatus ConnectTime                            Timeout
Connected     Wed Aug 25 10:06:04 2010 (1282745164) 120

PeerWindowEnd                         PeerWindow
Wed Aug 25 10:12:35 2010 (1282745555) 300

LocalHost                               LocalService
spock2                                  55555

RemoteHost                              RemoteService      RemoteInstance
spock1                                  55555              db2inst

PrimaryFile  PrimaryPg  PrimaryLSN
S0000000.LOG 0          0x0000000001388000

StandByFile  StandByPg  StandByLSN         StandByRcvBufUsed
S0000000.LOG 0          0x0000000001388000 0%
```

You should see output similar to the following lines on the primary node (e.g., spock1):

```
C:\Program Files\IBM\SQLLIB\BIN>db2pd -hadr -db hadrdb

Database Partition 0 -- Database HADRDB -- Active -- Up 0 days 00:02:34 -- Date
08/25/2010 10:07:48

HADR Information:
Role    State               SyncMode HeartBeatsMissed   LogGapRunAvg (bytes)
Primary Peer                Sync     0                  0

ConnectStatus ConnectTime                            Timeout
Connected     Wed Aug 25 10:06:04 2010 (1282745164) 120

PeerWindowEnd                         PeerWindow
Wed Aug 25 10:12:35 2010 (1282745555) 300

LocalHost                               LocalService
spock1                                  55555

RemoteHost                              RemoteService      RemoteInstance
spock2                                  55555              db2inst

PrimaryFile  PrimaryPg  PrimaryLSN
S0000000.LOG 0          0x0000000001388000

StandByFile  StandByPg  StandByLSN
S0000000.LOG 0          0x0000000001388000
```

4.6 Register HADR with SA MP for Automatic Management

In this step, we will enable SA MP to monitor and manage the HADR pair automatically. We will do this by registering the HADR pair as a resource group with SA MP. *Do not* manually issue DB2 "takeover" commands after registering HADR as a resource group with SA MP.

Create HADR Resources Group and Resources

Before running this script, verify that the names in the definition files and mk script are customized to your environment. Refer to Appendix C.

Additionally, verify that the system is configured to reboot after receiving a stop error on a blue screen, with the message "*** Fatal System Error:…". This is defined within the "Startup and Recovery" dialog of the advanced section of the system properties. An example of configuring the nodes is through "System Properties" > "Advanced Tab" > "Startup and Recovery" > "Settings". Configuration changes need to be specified separately on both nodes.

From the "IBM Tivoli System Automation – Shell" window (e.g., **Start > IBM Tivoli System Automation – Shell**), run the following commands:

```
(P) $ cd /usr/sbin/rsct/sapolicies/db2
(P) $ ./mkhadr
```

When mkdb2 and mkhadr files run, verify that the commands are run successfully. Issue the **lssam** command to observe the output below.

```
$ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs
               |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip
               |- Online IBM.ServiceIP:db2ip:spock1
               '- Offline IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

5. Post Configuration Testing

Once the setup is complete, we can test our automated HADR environment. Issue the **lssam** command, and observe the output displayed to the screen. You will see output similar to this:

```
$ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs
               |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip
               |- Online IBM.ServiceIP:db2ip:spock1
               '- Offline IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

Below is a brief description of the resources shown in the preceding SA MPle output and what they represent:

1) Primary DB2 instance resource group:
   db2_db2inst_spock1_0-rg

   Member Resources:
   db2_db2inst_spock1_0-rs (primary DB2 instance)

2) Standby DB2 instance resource group:
   db2_db2inst_spock2_0-rg

   Member Resources:
   db2_db2inst_spock2_0-rs (standby DB2 instance)

3) HADR database resource group:
   db2_db2inst_db2inst_HADRDB-rg

   Member Resources:
   db2_db2inst_db2inst_HADRDB-rs (HADR DB)

The resource groups mentioned above are created for both the HADR configurations discussed in this paper. However, the created networks are different.

In the case of the single network HADR configuration setup, only the following equivalencies are created by mkdb2:

Displaying Equivalencies:
   virpubnic_spock1_spock2

In the following steps, we will go through simulating various failure scenarios, and see how the preceding system configuration reacts to such failures. You can assume that the system reaction to a failure scenario is identical for both HADR configurations, unless otherwise mentioned.

Before continuing with this section, you must note some key points:

For all of the following test cases, it is assumed that hadrdb is primary on spock1, and all of the instance resource groups are online.

The following lines explain the meaning of the states that you see after issuing the **`lssam`** command:

For instance resources:
- Online = instance is up
- Offline = instance is down
- Unknown = the state of the instance is unknown

For HADR resources:
- Online = HADR database is primary at the node name
- Offline = HADR database is standby at the node name
- Unknown = HADR database is not in peer state ***or*** HADR database is down

Note: If the HADR group is locked (or suspended from automation), that likely means HADR DB is not in Peer state.

6. Testing Topology Response to Common Failures

All commands are to be run from the "IBM Tivoli System Automation – Shell" window unless specified otherwise (e.g., **Start > IBM Tivoli System Automation – Shell**).

6.1 Controlled Failover Testing

1. As user1, move "db2_db2inst_db2inst_HADRDB-rs" resource from the primary node (e.g., spock1) to the standby node (e.g., spock2); in other words, perform a controlled failover as follows (ignore the "token" message):

   ```
   (P) $ rgreq –o move db2_db2inst_db2inst_HADRDB-rg
   ```

2. As user1, verify that the primary database (e.g., HADRDB) has successfully failed over to the standby node (e.g., spock2) as follows:

   ```
   (P) $ lssam
   ```

   Output similar to the following lines should be seen:

   ```
   (P) $ lssam
   Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
         |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs
                  |- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
                  '- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
         '- Online IBM.ServiceIP:db2ip
                  |- Offline IBM.ServiceIP:db2ip:spock1
                  '- Online IBM.ServiceIP:db2ip:spock2
   Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
         '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
   Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
         '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
   Online IBM.Equivalency:virpubnic_spock1_spock2
         |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
         '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
   ```

3. Return hadrdb back to being primary on spock1 as follows (ignore the "token" message):

   ```
   (P) $ rgreq –o move db2_db2inst_db2inst_HADRDB-rg
   ```

6.2 Testing Instance Failure: Primary Instance (e.g., db2inst)

1. As primary instance owner, simulate a primary instance failure as follows:

   From the db2cmd window:

   ```
   (P) % db2_kill
   ```

2. On the primary or standby node (e.g., spock2), issue the following command repeatedly until you see output similar to what you saw the first time you ran the command (i.e., db2_db2inst_spock1_0-rg is Online again):

```
(S) $ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs
               |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip
               |- Online IBM.ServiceIP:db2ip:spock1
               '- Offline IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

Note: It may take up to 2 minutes to see the instance restarted automatically.

6.3 Testing Instance Failure: Standby Instance (e.g., db2inst)

1. As standby instance owner, simulate a standby instance failure as follows:

   From the db2cmd window:

```
(S) db2_kill
```

2. On the standby node (e.g., spock2), issue **lssam** repeatedly until you see that db2_db2inst_spock2_0-rg is Online again, and issue **db2 activate db hadrdb** on the db2cmd command prompt to get the HADR database back to Peer state.

```
(S) $ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs Request=Lock
               |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip Control=SuspendedPropagated
               |- Online IBM.ServiceIP:db2ip:spock1
               '- Offline IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Pending online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

   From the db2cmd window:

```
(S) db2 activate db hadrdb
```

From the "IBM Tivoli System Automation – Shell" window:

```
(S) $ lssam

Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
     |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs
           |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
           '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
     '- Online IBM.ServiceIP:db2ip
           |- Online IBM.ServiceIP:db2ip:spock1
           '- Offline IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
     '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
     '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
     |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
     '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

Note: It may take up to 2 minutes to see the instance restarted automatically.

6.4 Testing Resource Group Failure: Primary Instance Resource Group

1. As root, bring the primary instance resource group (e.g., db2_db2inst_spock1_0-rg) offline as follows:

```
(P) $ chrg -o offline db2_db2inst_spock1_0-rg
```

2. Issue the following command and observe that the primary instance resource group goes offline:

```
(P) $ lssam

Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Unknown IBM.Application:db2_db2inst_db2inst_HADRDB-rs
                |- Unknown IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
                '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip
                |- Online IBM.ServiceIP:db2ip:spock1
                '- Offline IBM.ServiceIP:db2ip:spock2
Offline IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Offline
      '- Offline IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

3. Now bring the primary instance resource group back online by issuing the following command:

```
(P) $ chrg -o online db2_db2inst_spock1_0-rg
```

4. Verify that the primary instance resource group is back online as follows:

```
(P) $ lssam
```

6.5 Testing Resource Group Failure: Standby Instance Resource Group

1. Bring the standby instance resource group (e.g., db2_db2inst_spock2_0-rg) offline as follows:

```
(S) $ chrg -o offline db2_db2inst_spock2_0-rg
```

2. Issue the following command and observe that the standby instance resource group goes offline:

```
(S) $ lssam

Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs Request=Lock
               |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip Control=SuspendedPropagated
               |- Online IBM.ServiceIP:db2ip:spock1
               '- Offline IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Offline IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Offline
      '- Offline IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

3. Bring the standby instance resource group back online as follows:

```
(S) $ chrg -o online db2_db2inst_spock2_0-rg
```

4. Verify that the standby instance resource group has been restarted successfully:

```
(S) $ lssam
```

5. **(*)** You need to manually activate the database on the

   Standby: From the db2cmd window:

```
(S)   db2 activate db HADRDB
```

6. Verify that the HADR database is back in Peer mode.

(*) This step is not required if you implemented automatic HADR reintegration. For more information, see Appendix D.

6.6 Testing Network Adapter Failure (e.g., Local Area Connection 1)

1. Pull the cable on the NIC that is currently hosting the primary instance's IP address (e.g., Local Area Connection 1). The system behavior will be identical to that described in the "Node Failure" test that follows this one on the standby node. (**Note:** Allow enough time for the old standby node (spock2) to switch to the new primary node):

```
(S) $ lssam

$ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs Request=Lock
               |- Failed offline IBM.Application:db2_db2inst_db2inst_HADRDB-
rs:spock1 Node=Offline
               '- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip Control=SuspendedPropagated
               |- Failed offline IBM.ServiceIP:db2ip:spock1 Node=Offline
               '- Online IBM.ServiceIP:db2ip:spock2
Failed offline IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Failed offline IBM.Application:db2_db2inst_spock1_0-rs:spock1
Node=Offline Binding=Unbindable
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Offline IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-
10B0D743436B:spock1 Node=Offline
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

Db2pd output on spock2

From the db2cmd window:

```
C:\Program Files\IBM\SQLLIB\BIN>db2pd -hadr -alldbs

Database Partition 0 -- Database HADRDB -- Active -- Up 0 days 00:22:21 -- Dat
e 08/25/2010 18:05:06

HADR Information:
Role    State               SyncMode HeartBeatsMissed   LogGapRunAvg (bytes)
Primary Disconnected        Sync     0                  0

ConnectStatus ConnectTime                           Timeout
Disconnected  Wed Aug 25 18:04:19 2010 (1282773859) 120

PeerWindowEnd                       PeerWindow
Null (0)                            300

LocalHost                              LocalService
spock2                                 55555

RemoteHost                             RemoteService      RemoteInstance
spock1                                 55555              db2inst

PrimaryFile  PrimaryPg  PrimaryLSN
S0000002.LOG 0          0x0000000001B88000

StandByFile  StandByPg  StandByLSN
S0000000.LOG 0          0x0000000000000000
```

2. Plug the network cable back to the old primary system (e.g., spock1). If the database in the old primary machine is still activated, issue **db2_kill** before attempting reintegration.

   From the db2cmd window:

   ```
   db2_kill
   ```

   Important: If the old primary database is already active, deactivating it may lead to later reintegration failure. You must stop the old primary instance using **db2_kill** or by ending the db2sysc.exe process manually.

   If reintegration of the HADR pair fails, the standby database may have to be re-established via a backup image of the current primary database.

3. **(*)** Once the old primary instance is back Online, you can now re-establish the HADR pair from the old primary machine (e.g., spock1):

   From the db2cmd window, as the original primary instance owner, db2inst, issue:

   ```
   db2 start hadr on db hadrdb as standby
   ```

4. The HADR pair should now be re-established (you can issue the **lssam** command to check). To bring the primary database back to spock1, issue the following command (ignore the "token" message):

   ```
   (P) $ rgreq -o move db2_db2inst_db2inst_HADRDB-rg
   ```

5. Verify that topology has returned to its original state before the cable was pulled:

   ```
   (P) $ lssam
   ```

(*) This step is not required if you implemented automatic HADR reintegration. For more information, see Appendix D.

6.7 Node Failure

For this test case to work, the script will perform a **takeover by force** command if the HADR pair drops out of Peer state before SA MP can issue a failover to the standby database. **Important:** If HADR is not operating in synchronization mode (mode = sync), the standby database may take over as primary at a time when it is not in sync with the primary database that failed. If this is the case, then later reintegration of the HADR pair may fail, and the standby database may have to be re-established using a backup image of the current primary database:

Primary Node Failure (*)

(*) Steps 2 - 6 are not required if you implemented automatic HADR reintegration. For more information, see Appendix D.

1. First, verify that the status of all resources is normal as follows:

   ```
   (P) $ lssam
   ```

Output similar to the following lines should be seen:

```
$ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
     |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs
                |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
                '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
     '- Online IBM.ServiceIP:db2ip
                |- Online IBM.ServiceIP:db2ip:spock1
                '- Offline IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
     '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
     '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
     |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
     '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

2. Simulate a failure of the primary node (e.g., spock1) by rebooting or shutdown windows.

3. Verify status of recovery by issuing the following command repeatedly:

```
(S)  $ lssam
```

After a few minutes, output similar to the following lines should be seen:

```
$ lssam
Pending online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
     |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs Request=Lock
                |- Failed offline IBM.Application:db2_db2inst_db2inst_HADRDB-
rs:spock1 Node=Offline
                '- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
     '- Online IBM.ServiceIP:db2ip Control=SuspendedPropagated
                |- Failed offline IBM.ServiceIP:db2ip:spock1 Node=Offline
                '- Online IBM.ServiceIP:db2ip:spock2
Failed offline IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
     '- Failed offline IBM.Application:db2_db2inst_spock1_0-rs:spock1
Node=Offline Binding=Unbindable
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
     '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
     |- Offline IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
Node=Offline
     '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

Verify that the HADR database is now primary on node spock2 using the **`db2pd -hadr -db hadrdb`** command.

From the db2cmd window:

```
C:\Program Files\IBM\SQLLIB\BIN>db2pd -hadr -db HADRDB

Database Partition 0 -- Database HADRDB -- Active -- Up 0 days 00:1e 08/25/2010
19:02:18

HADR Information:
Role    State               SyncMode HeartBeatsMissed   LogGapRunAvg
Primary Disconnected        Sync     0                  0

ConnectStatus ConnectTime                           Timeout
Disconnected  Wed Aug 25 18:58:23 2010 (1282787903) 120

PeerWindowEnd                       PeerWindow
Null (0)                            300

LocalHost                              LocalService
spock2                                 55555

RemoteHost                             RemoteService      RemoteIns
spock1                                 55555              db2inst

PrimaryFile  PrimaryPg  PrimaryLSN
S0000002.LOG 0          0x0000000001B88000

StandByFile  StandByPg  StandByLSN
S0000000.LOG 0          0x0000000000000000
```

4. Once the old primary machine (i.e., spock1) comes back online, lssam output will be as follows:

```
$ lssam
Pending online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Request=Move No
minal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs Request=Lock
               |- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip Control=SuspendedPropagated
               |- Offline IBM.ServiceIP:db2ip:spock1
               '- Online IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

5. Verify that the database is not active on the old primary machine (i.e., spock1):

   From the db2cmd window:

```
C:\Program Files\IBM\SQLLIB\BIN>db2pd -hadr -alldbs

Option -hadr requires -db <database> or -alldbs option and active database.
```

Important: If the old primary database is already active, deactivating it may lead to later reintegration failure. You must stop the old primary instance using **db2_kill** or by ending the db2sysc.exe process manually, and then proceed to step 6.

If reintegration of the HADR pair fails, the standby database may have to be re-established using a backup image of the current primary database.

6. You can now re-establish the HADR pair from the old primary machine (i.e., spock1).

   From the db2cmd window, as the original primary instance owner, db2inst, issue:

```
C:\Program Files\IBM\SQLLIB\BIN>db2 start hadr on db hadrdb as standby

DB20000I  The START HADR ON DATABASE command completed successfully.
```

7. The HADR pair should now be re-established (you can issue the **lssam** command to check). To bring the primary database back to spock1, issue the following command (ignore the "token" message):

```
(P) $ rgreq -o move db2_db2inst_db2inst_HADRDB-rg
```

8. Verify that HADR has returned to its original state before the primary node failure:

```
(P) $ lssam
```

Standby Node Failure (*)

(*) Steps 2 - 4 are not required if you implemented automatic HADR reintegration. For more information, see Appendix D.

1. First, verify that the status of all resources is normal as follows:

```
(P) $ lssam
```

   Output similar to the following lines should be seen:

```
$ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs
               |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip
               |- Online IBM.ServiceIP:db2ip:spock1
               '- Offline IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

2. Simulate a failure of the standby node (e.g., spock2) by rebooting or shutdown windows.

3. Check the status of recovery by issuing the following command repeatedly:

(P) `$ lssam`

After a few minutes, output similar to the following lines should be seen:

```
$ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs Request=Lock
               |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Failed offline IBM.Application:db2_db2inst_db2inst_HADRDB-
rs:spock2 Node=Offline
      '- Online IBM.ServiceIP:db2ip Control=SuspendedPropagated
               |- Online IBM.ServiceIP:db2ip:spock1
               '- Failed offline IBM.ServiceIP:db2ip:spock2 Node=Offline
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Failed offline IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Failed offline IBM.Application:db2_db2inst_spock2_0-rs:spock2
Node=Offline Binding=Unbindable
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Offline IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-
43B651541F1F:spock2 Node=Offline
```

Verify that the HADR database is still primary on node spock1 using the **`db2pd –hadr –db hadrdb`** command.

From the db2cmd window:

```
C:\Program Files\IBM\SQLLIB\BIN>db2pd -hadr –db HADRDB

Database Partition 0 -- Database HADRDB -- Active -- Up 0 days 01:26:06 -- Date
09/13/2010 17:10:21

HADR Information:
Role    State               SyncMode HeartBeatsMissed   LogGapRunAvg (bytes)
Primary Disconnected        Sync     0                  0

ConnectStatus ConnectTime                           Timeout
Disconnected  Mon Sep 13 16:47:51 2010 (1284421671) 120

PeerWindowEnd                          PeerWindow
Mon Sep 13 16:52:27 2010 (1284421947) 300

LocalHost                                 LocalService
spock1                                    55555

RemoteHost                                RemoteService      RemoteInstance
spock2                                    55555              db2inst

PrimaryFile  PrimaryPg  PrimaryLSN
S0000002.LOG 0          0x0000000001B88000

StandByFile  StandByPg  StandByLSN
S0000002.LOG 0          0x0000000001B88000
```

4. Once the standby machine (i.e., spock2) comes back online, you can re-establish the HADR pair as follows:

   As the original standby instance owner, db2inst:
   spock2: > db2 activate db hadrdb

   From the db2cmd window:

```
C:\Program Files\IBM\SQLLIB\BIN>db2 activate db hadrdb

DB20000I  The ACTIVATE DATABASE command completed successfully.
```

5. Check that HADR has returned to its original state before the primary node failure:

   (P) $ lssam

7. Conclusion

This paper guided you through the implementation of an automated failover solution for the IBM® DB2® Enterprise Server Edition for Linux, UNIX, and Windows Version product. The solution is based on a combination of the HADR feature and the IBM Tivoli® SA MP product. The setup that is described in this paper focuses on the Windows operating system and was updated to reflect enhancements that were done in DB2 9.7 Fix Pack 8.

Appendix A: Understanding How SA MP Works

IBM Tivoli System Automation for Multiplatforms (IBM Tivoli SA MP) provides a framework to automatically manage the availability of what are known as resources.

Examples of resources are:

- Any piece of software for which start, monitor, and stop scripts can be written to control.
- Any network interface card (NIC) to which SA MP has been granted access. That is, SA MP will manage the availability of any IP address that a user wants to use by floating that IP address among NICs that it has been granted access to.

For example, both a DB2 instance and an HADR database itself have start, stop, and monitor commands. Therefore, SA MP scripts can be written to manage these resources automatically. In fact, you can create the scripts by copying them from Appendix C as user1 (root) after installing DB2 9.5:

```
(P)(S) # cd /usr/sbin/rsct/sapolicies/db2/
```

If directories "sapolicies" and "db2" do not exist, create them.

```
$ ls
db2.def       hadr_monitor.ksh    mkdb2         rmdb2
db2ip.def     hadr_start.ksh      mkhadr
hadr.def      hadr_stop.ksh       README
```

Scripts, as well as other attributes of a resource, are needed by SA MP to manage that resource. SA MP stores a resource's attributes in an object container, much like the attributes of a Java class. In fact, SA MP manages a resource by instantiating a class for that resource.

Examples of classes that SA MP instantiates to manage different resources are*:

- IBM.Application – a resource class for applications (e.g., DB2 instance)
- IBM.ServiceIP – a resource class that has special attributes to define an IP address and a net mask (e.g., IP address of a DB2 instance)
- IBM.Equivalency – a resource class that defines equivalent NICs to host an HA IP address (e.g., Local Network Connection and Local Network Connection 2 could be made equivalent to host the IP address of primary DB2 instance).

*For more information on these and other resource classes, refer to http://www.ibm.com/software/tivoli/products/sys-auto-linux/.

SA MP also allows related resources to be managed in what are known as resource groups. With SA MP, you can specify that all resources within a given resource group will be online at one and only one physical node at any point in time. Also, all of those resources will reside on the same physical node.

Examples of resource groups (i.e., related resources) are:

- A DB2 instance, its IP address, and all of the databases that it manages (e.g., HADRDB)

Finally, SA MP provides high availability (HA) for any resource group that it manages, by restarting all of its resources if it fails. The resource group will be restarted on an appropriate node in the currently online cluster domain. An appropriate node must contain a copy of all of the resources that are defined in the failing resource group, to be selected as a node to restart on.

The following examples show "dialogs" that would occur between SA MP nodes in Fig. 1, Typical HADR Environment with SA MP, in the event of various failures/user actions.

Note: For each dialog, assume that spock1 is the primary database node and spock2 is the standby database node:

Standby Node Loses Network Communication to Primary Node:

SA MP on spock2: Hey, SA MP on TieBreaker IP, I can't talk to SA MP on spock1 anymore. Is it OK for me to tell hadrdb, managed by standby instance, to become primary?

SA MP on spock3: No, I am still receiving a heartbeat from SA MP on spock1, so hadrdb managed by standby instance cannot become primary.

SA MP on spock2: OK then.

Primary Node Fails:

SA MP on spock2: Hey, SA MP on TieBreaker IP, I can't talk to SA MP on spock1 anymore. Is it OK for me to tell hadrdb, managed by standby instance, to become primary?

SA MP on spock3: Yes, I am not receiving a heartbeat from SA MP on spock1 either, so hadrdb managed by standby instance can become primary. Don't worry, if spock1 comes back online. I will make sure that DB2 is not started so that we do not have two primary databases.

SA MP on spock2: OK then. Hey, standby instance, have hadrdb take over by force as primary.

DB2: Done.

Switch HADR Roles Manually:

User types on spock1: rgreq –o move db2_db2inst_db2inst_HADRDB-rg

SA MP on spock1: OK, I will move the HADR resource group (i.e., db2_db2inst_db2inst_HADRDB-rg) off spock1 and onto spock2. I will shut down the resource group now. Hey, standby instance, make hadrdb switch roles. At the same time, I want you SA MP on spock2 to bring online the resource group called db2_db2inst_db2inst_HADRDB-rg.

DB2: OK, done.

SA MP on spock2: OK, done.

Appendix B: Troubleshooting Tips

1. Text editor:

   While working under the Subsystem for UNIX-based Applications (SUA) environment, you should always use a UNIX editor to modify the files (e.g., "vi" editor, which comes with the "Utilities and SDK for UNIX-based Application_X86" package" for SUA).

   Alternatively, if you need to work with files in a Windows environment, you must convert them to adopt the UNIX convention for line endings. You can run the following command to convert your text file from Windows to UNIX format:

   From the "IBM Tivoli System Automation – Shell" window:

   ```
   flip –u <filename>
   ```

2. Requirements on input names:

   - Instance names for the primary and standby databases must be the same.
   - The case and format for resource names must be consistent throughout. For example, the host name must be the same case as the 'hostname' return. For database names, the common practice is to use all capital letters.

3. Non-default DB2 install path:

   If the default DB2 install path is not used, or it does not exist in the PATH environment variable, you must change the PATH variable at the beginning of each hadr script (hadr_monitor.ksh, hadr_start.ksh, and hadr_stop.ksh).

   For example, if DB2 is installed on this location D:\Program Files\IBM\SQLLIB_TEST, then you will need to modify this line:

   ```
   PATH=/bin:/usr/bin:/sbin:/usr/sbin:/usr/sbin/rsct/bin:"/dev/fs/C/Program
   Files/IBM/SQLLIB/BIN":/dev/fs/C/WINDOWS/system32
   ```

   To:

   ```
   PATH=/bin:/usr/bin:/sbin:/usr/sbin:/usr/sbin/rsct/bin:"/dev/fs/D/Program
   Files/IBM/SQLLIB_TEST/BIN":/dev/fs/C/WINDOWS/system32
   ```

4. Non-default WINDOWS location:

   If your operating system is not installed on the default C drive, C:/WINDOWS location, you must update the PATH variable to include it. (e.g. In the step above, you will change the default "/dev/fs/C/WINDOWS/system32" to "/dev/fs/D/WINDOWS/system32" where D is the new drive) In addition, you will also need to modify the ping.exe location inside the samtb_net file.

   In the SUA environment, this file is located here: /usr/sbin/rsct/bin/samtb_net
   Or in the Windows environment: <D:\WINDOWS>\SUA\usr\sbin\rsct\bin\samtb_net

   Modify this line to the correct WINDOWS location:

   ```
   $retc = system("/dev/fs/C/WINDOWS/system32/ping.exe -w 5000 -n 1 $hos
   t >/dev/null 2>&1");
   ```

5. Resources not created after running mkdb2:

   Always verify the output of the setup script run. Confirm that the data input files are updated correctly. Here are some common errors you may see from the mkdb2 run:

   Problem 1:

```
+ mkequ virpubnic_spock1_spock2 IBM.NetworkInterface:A13D316B-A16B-4D34-84D3-
10B0D743436B:spock1,78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
mkequ: 2622-014 The resource "A13D316B-A16B-4D34-84D3-10B0D743436B" not found
in the class "IBM.NetworkInterface".
```

   Solution 1:

   Verify that the NIC name is correct. The NIC names can be found by issuing the following in a "IBM Tivoli System Automation – Shell" window:

```
$ lsrsrc IBM.NetworkInterface Name NodeNameList
Resource Persistent Attributes for IBM.NetworkInterface
resource 1:
   Name         =  ”78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F”
   NodeNameList =  {“spock2”}
resource 2:
   Name         =  “B13D316B-A16B-4D34-84D3-10B0D743436B”
   NodeNameList =  {“spock1”}
```

   This error can be ignored if you will not be defining a Service IP address for the HADR database.

   Problem 2:

```
+ mkrsrc -f db2.def IBM.Application
mkrsrc: 2619-004 PersistentResourceAttributes not found in input file
db2.def.
mkrsrc: 2612-083 Error processing file db2.def for input.
```

   Solution 2:

   Verify that the "PersistentResourceAttributes" header does exist. This error may also be caused by Windows line-ending characters in the db2.def (e.g., if the file has been previously edited using Windows editor). Using "vi", you will see the ^M characters appended at the end of each line. See tip #1 for resolution.

Appendix C: Resources and Resource Groups Setup Scripts

These scripts can be found under the <install_path>\sqllib\SA MPles\tsa directory where <install_path> is the location of DB2 on your hard drive. The default location for <install_path> is C:\Program Files\IBM on Windows.

In the "IBM Tivoli System Automation - Shell" window, copy the SA MPle files to the /usr/sbin/rsct/sapolicies/db2 directory. Create this directory if it does not already exist. Alternatively, from Windows Explorer, copy the files onto this default location: C:\WINDOWS\SUA\usr\sbin\rsct\sapolicies\db2

Definition files

```
*******************************************************************************
* db2.def     SA MPle data input file for a DB2 instance resource
*             The following attributes should be modified:
*             Name, StartCommand, StopCommand, MonitorCommand, NodeNameList,
*             UserName. In addition, the convention below must be followed:
*
*             Name       db2_<instance>_<host>_0-rs
*             *Command   /usr/sbin/rsct/bin/samservice -<x> <service_name>
*
*                        <instance> is the name of the database instance
*                        <host> is the primary hostname for resource 1, and the
*                           standby hostname for resource 2
*                        <service_name> is the name of the DB2 service that
*                           appears in the Windows Services console. Note that
*                           although your default instance name is "DB2", its
*                           service name may be "DB2-0". Verify this in here:
*                           Start > Administrative Tools > Services
*
* db2ip.def   SA MPle data input file for a virtual IP resource
*             The following attributes should be modified:
*             Name, NodeNameList, IPAddress, NetMask.
*
* hadr.def    SA MPle data input file for an HADR database resource
*             The following attributes should be modified:
*             Name, StartCommand, StopCommand, MonitorCommand, UserName,
*             NodeNameList. In addition, the convention below must be followed:
*
*             Name       db2_<instance>_<instance>_<DBname>-rs
*             *Command   /usr/sbin/rsct/sapolicies/db2/hadr_<x>.ksh <instance>
*                        <instance> <DBname>
*
*                        <instance> is the name of the database instance
*                        <DBname> is the HADR database name, capitalized
*
*******************************************************************************
```

$ cat db2.def

```
PersistentResourceAttributes::
resource 1:
        Name                    = "db2_db2inst_spock1_0-rs"
        StartCommand            = "/usr/sbin/rsct/bin/samservice -s db2inst"
        StopCommand             = "/usr/sbin/rsct/bin/samservice -p db2inst"
        MonitorCommand          = "/usr/sbin/rsct/bin/samservice -m db2inst"
        MonitorCommandPeriod    = 30
        MonitorCommandTimeout   = 180
        NodeNameList            = {'spock1'}
```

```
        StartCommandTimeout   = 300
        StopCommandTimeout    = 300
        UserName              = "user1"
        RunCommandsSync       = 1
        ResourceType          = 0

resource 2:
        Name                  = "db2_db2inst_spock2_0-rs"
        StartCommand          = "/usr/sbin/rsct/bin/samservice -s db2inst"
        StopCommand           = "/usr/sbin/rsct/bin/samservice -p db2inst"
        MonitorCommand        = "/usr/sbin/rsct/bin/samservice -m db2inst"
        MonitorCommandPeriod  = 30
        MonitorCommandTimeout = 180
        NodeNameList          = {'spock2'}
        StartCommandTimeout   = 300
        StopCommandTimeout    = 300
        UserName              = "user1"
        RunCommandsSync       = 1
        ResourceType          = 0
```

$ cat db2ip.def

```
PersistentResourceAttributes::
resource 1:
        Name           = "db2ip"
        NodeNameList   = {'spock1','spock2'}
        IPAddress      = '9.26.124.34'
        NetMask        = '255.255.254.0'
        ResourceType   = 1
        ProtectionMode = 1
```

$ cat hadr.def

```
PersistentResourceAttributes::
resource 1:
        Name                  = "db2_db2inst_db2inst_HADRDB-rs"
        ResourceType          = 1
        StartCommand          = "/usr/sbin/rsct/sapolicies/db2/hadr_start.ksh db2inst
db2inst HADRDB"
        StopCommand           = "/usr/sbin/rsct/sapolicies/db2/hadr_stop.ksh db2inst
db2inst HADRDB"
        MonitorCommand        = "/usr/sbin/rsct/sapolicies/db2/hadr_monitor.ksh db2inst
db2inst HADRDB"
        MonitorCommandPeriod  = 20
        MonitorCommandTimeout = 120
        StartCommandTimeout   = 300
        StopCommandTimeout    = 15
        UserName              = "user1"
        RunCommandsSync       = 1
        ProtectionMode        = 0
        NodeNameList          = {"spock1","spock2"}
```

Resource and resource group setup scripts

```
*******************************************************************************
* mkdb2       Creates a DB2 resource based on db2.def
* mkhadr      Creates an HADR database resource based on hadr.def, and
*             Service IP resource based on db2ip.def
* rmdb2       Removes any clustered resources and removes the cluster domain
*
*             These scripts should be modified to suit your environment.
*             Here are a few entries that require customization:
*
*             DBNAME         HADR database name, capitalized
*             HADRINSTANCE   DB2 instance name (default is "DB2")
*             HOST1 / HOST2  The primary and standby hostname
*             NIC1  / NIC2   The name of the NIC resource from primary and
*                            standby instance. These names can be obtained by
*                            running SA MP command below:
*                            "lsrsrc IBM.NetworkInterface Name NodeNameList"
*                            Note: This is only required if you use a Service
*                            IP address to connect to the HADR database
*             TIEBREAKER     network address that serves as a tiebreaker
*
*******************************************************************************
```

$ cat mkdb2

```
#!/bin/ksh
set -x

# Modify the entries here:

HOST1="spock1"
HOST2="spock2"
HADRINSTANCE="db2inst"
NIC1="B13D316B-A16B-4D34-84D3-10B0D743436B"
NIC2="78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F"

################################################

# Starting setup...

echo "Making equivalency ..." mkequ
virpubnic_${HOST1}_${HOST2}
IBM.NetworkInterface:${NIC1}:${HOST1},${NIC2}:${HOST2}
sleep 10

echo "Making resource (rs)"
mkrsrc -f db2.def IBM.Application

sleep 30

echo "Making resource group (rg) ..."
mkrg db2_${HADRINSTANCE}_${HOST1}_0-rg  db2_${HADRINSTANCE}_${HOST2}_0-rg
sleep 10

echo "Adding members to rg ..."
addrgmbr -g db2_${HADRINSTANCE}_${HOST1}_0-rg
IBM.Application:db2_${HADRINSTANCE}_${HOST1}_0-rs:${HOST1}
sleep 10
addrgmbr -g db2_${HADRINSTANCE}_${HOST2}_0-rg
IBM.Application:db2_${HADRINSTANCE}_${HOST2}_0-rs:${HOST2}

sleep 30
```

```
lsrg -m
lsrsrc IBM.Application Name NodeNameList OpState
date
lssam
date
sleep 10

chrg -o online -s "Name like '%'"

sleep 120

date
lssam
date
```

$ cat mkhadr

```
#!/bin/ksh
set -x

# Modify the entries here:

HADRINSTANCE="db2inst"
DBNAME="HADRDB"
TIEBREAKER="9.26.124.30"

#########################################################

# Computing resource names...

RGNAME=db2_${HADRINSTANCE}_${HADRINSTANCE}_${DBNAME}-rg
RSNAME=db2_${HADRINSTANCE}_${HADRINSTANCE}_${DBNAME}-rs

# Starting setup...

echo "Making HADR resources ..."
export CT_MANAGEMENT_SCOPE=2

mkrsrc -f hadr.def IBM.Application
sleep 20

echo "Making IP resources ..."
mkrsrc -f db2ip.def IBM.ServiceIP
sleep 20

echo "Making resource group (rg) ..."
mkrg ${RGNAME}
sleep 20

echo "Adding membes to rg ..."
addrgmbr -g ${RGNAME} IBM.Application:${RSNAME}
sleep 20


mkrsrc IBM.TieBreaker Type="EXEC" Name="mynetworktb"
DeviceInfo='PATHNAME=/usr/sbin/rsct/bin/samtb_net Address='${TIEBREAKER}' Log=1'
PostReserveWaitTime=30

sleep 60

chrg -o online ${RGNAME}

sleep 60
```

```
echo "Making relationships ..."
lsrsrc IBM.TieBreaker
lsrsrc -c IBM.PeerNode
chrsrc -c IBM.PeerNode OpQuorumTieBreaker="mynetworktb"
chrsrc -c IBM.PeerNode CritRsrcProtMethod=1
lsrsrc -c IBM.PeerNode
```

$ cat rmdb2

```
#!/bin/ksh
set -x

# Modify the entries here:

HOST1="spock1"
HOST2="spock2"
HADRINSTANCE="db2inst"
DBNAME="HADRDB"
DOMAINNAME="hadr_domain"

############################################################

# Computing names...

RGNAME1=db2_${HADRINSTANCE}_${HOST1}_0-rg
RGNAME2=db2_${HADRINSTANCE}_${HOST2}_0-rg
DBRGNAME=db2_${HADRINSTANCE}_${HADRINSTANCE}_${DBNAME}-rg

# Removing setup...

chrg -o offline -s "Name like '%'"
sleep 30

echo "Removing resource groups..."
rmrg ${RGNAME2} ${RGNAME1} ${DBRGNAME}

sleep 10

echo "Removing equivalencies..."
rmequ virpubnic_${HOST1}_${HOST2}

sleep 10

echo "Remove ressources ..."
rmrsrc -s 'Name like "db2%"' IBM.Application
rmrsrc -s 'Name like "db2%"' IBM.ServiceIP
rmrsrc -s 'Name like "db2_${HADRINSTANCE}_${HADRINSTANCE}_%"' IBM.Application
sleep 30

lsequ
lsrsrc IBM.Application Name NodeNameList OpState
lsrsrc IBM.ServiceIP Name NodeNameList OpState

lssam
lsrpdomain

stoprpdomain -f ${DOMAINNAME}
sleep 30

rmrpdomain -f ${DOMAINNAME}
sleep 30

lsrpnode
```

Resource management scripts

```
*******************************************************************************
*
* hadr_monitor.ksh   Monitors an HADR database resource and checks for its
*                    status
* hadr_start.ksh     Starts up the HADR database resource
* hadr_stop.ksh      Stops an HADR database resource
*
*******************************************************************************
```

$ cat hadr_monitor.ksh

```
#!/bin/ksh -p
#-------------------------------------------------------------------------
# (C) COPYRIGHT International Business Machines Corp. 2001-2010
# All Rights Reserved
#
# US Government Users Restricted Rights - Use, duplication or
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
#
# NAME: hadr_monitor.ksh
#
# (%W%) %E%  %U%
#
# FUNCTION: Probe for specified HADR pair alive
#
# INPUT:  hadr_monitor.ksh db2instp db2insts db2hadrdb verbose [S]
#
# OUTPUT: see probehadr() function for description of return codes
#
#-------------------------------------------------------------------------
PATH=/bin:/usr/bin:/sbin:/usr/sbin:/usr/sbin/rsct/bin:"/dev/fs/C/Program
Files/IBM/SQLLIB/BIN"
export PATH=$PATH:.

RESOURCE_METHOD=probe
PROGNAME=$(basename $0)
PROGPATH=$(dirname $0)
RT_BASEDIR=$PROGPATH
DB2HADRINSTANCE1=${1?}
DB2HADRINSTANCE2=${2?}
DB2HADRDBNAME=${3?}
VERBOSE=${4:-verbose}
monitor_as_standby=${5:-N}

PROBE=$PROGNAME
START=
STOP=
ACTIVATE=
SVC_PROBE=${RT_BASEDIR}/${PROBE:-hadr_monitor.ksh}
START_CMD=$RT_BASEDIR/${START:-hadr_start.ksh}
STOP_CMD=$RT_BASEDIR/${STOP:-hadr_stop.ksh}

export CT_MANAGEMENT_SCOPE=2
export DB2INSTANCE=${DB2HADRINSTANCE1?}

if [[ "$VERBOSE" == "verbose" ]]; then
   typeset -ft $(typeset +f)
   set -x
else
   #  Close stdout   and   stderr
   exec  2> /dev/null
   exec  1> /dev/null
```

```
   set   +x
fi

grepFlags='-e'

CLUSTER_APP_LABEL=db2_${DB2HADRINSTANCE1}_${DB2HADRINSTANCE2}_${DB2HADRDBNAME?}

# Name the resource group and the resource
DATA_SVC_RG_NAME=${CLUSTER_APP_LABEL}-rg
DATA_SVC_R_NAME=${CLUSTER_APP_LABEL}-rs

temp_snap=/tmp/.${PROGNAME?}_${DB2HADRDBNAME?}_$$
temp_snap2=/.${PROGNAME?}_${DB2HADRDBNAME?}_$$

TRAPSIGNALS="TERM KILL"
# trap 'rm -f $temp_snap ; exit $rc' 0 1 2 3 6 9 15


############################################################
# resource_group_state_on_this_node()
#
# INPUT: ResourceClass, Resource, ResourceType, Node
# OUTPUT:rg_state [Online=1 || Offline]
#
############################################################
resource_group_state_on_this_node()
{
   OpState=$( lsrsrc-api -s ${ResourceClass}::'Name="'${Resource}'" '"${ResourceType}"'
&& NodeNameList={"'${Node}'"} '::OpState 2> /dev/null)

   rg_state=${OpState?}
}

############################################################
# set_candidate_P_instance()
#
# INPUT: DB2HADRINSTANCE1, DB2HADRINTANCE2
# OUTPUT:candidate_P_instance
#
############################################################
set_candidate_P_instance()
{
   Node=$(hostname | tr "." " " | awk '{print $1}')

   if [[ $DB2HADRINSTANCE1 == $DB2HADRINSTANCE2 ]]; then
      candidate_P_instance=$DB2HADRINSTANCE1
   else

      RSNAME1=db2_${DB2HADRINSTANCE1?}_
      RSNAME2=db2_${DB2HADRINSTANCE2?}_

      Resource=${RSNAME1?}
      ResourceClass=IBM.Application
      NodeRG1long=$(lsrsrc-api -s ${ResourceClass?}::'Name like
"'${Resource?}%'"'::NodeNameList | tr "{" " " | tr "}" " " | tr "." " " | awk '{print
$1}' | tail -1)
      NodeRG1=$(echo $NodeRG1long | tr "." " " | awk '{print $1}')

      Resource=${RSNAME2?}
      ResourceClass=IBM.Application
      NodeRG2long=$(lsrsrc-api -s ${ResourceClass?}::'Name like
"'${Resource?}%'"'::NodeNameList | tr "{" " " | tr "}" " " | tr "." " " | awk '{print
$1}' | tail -1)
      NodeRG2=$(echo $NodeRG2long | tr "." " " | awk '{print $1}')
```

```
      if   [[ $NodeRG1 == $Node && $NodeRG2 != $Node ]]; then
         candidate_P_instance=$DB2HADRINSTANCE1
      elif [[ $NodeRG1 != $Node && $NodeRG2 == $Node ]]; then
         candidate_P_instance=$DB2HADRINSTANCE2
      else
         # either both instances are up in resource groups
         # on this node, or neither is
         # figure out the one that wants to be primary

         rm -rf log.txt
         db2clpex.exe -c DB2 -z log.txt -tv "get db cfg for ${DB2HADRDBNAME?}"
         HADR_LOCAL_HOST1=cat log.txt | grep -e HADR_LOCAL_HOST | awk '{print $7}'
         HADR_LOCAL_HOST2=HADR_LOCAL_HOST2
         rm -rf log.txt


         if [[ $HADR_LOCAL_HOST1 == $Node ]]; then
            candidate_P_instance=$DB2HADRINSTANCE1
         elif [[ $HADR_LOCAL_HOST2 == $Node ]]; then
            candidate_P_instance=$DB2HADRINSTANCE2
         else
            candidate_P_instance=
         fi
      fi
   fi
}

############################################################
# probehadr_on_this_node()
#
# Probe status of HADR on the active resource on this node
# Return 4 if specified DB did not activate and activating
# Return 0 if specified DB2 HADR is unknown
# Return 1 if specified DB2 HADR is online as Primary,Peer on this node
# Return 3 if specified DB2 HADR is online as Primary,non-Peer on this node
# Return 2 if specified DB2 HADR is online as Standby,Peer on this node
# Return 40 if specified DB2 HADR is online as Standby,Not Peer on this node
#
############################################################
probehadr_on_this_node()
{
   # Use db2pd instead of SNAPSHOT if possible ...
   db2pd.exe -hadr -db ${DB2HADRDBNAME?} \
   | grep ${grepFlags?} "Sync " \
   | grep "[a-zA-Z]" \
   | awk '{print $1 "\n" $2}' > $temp_snap

   if [ -r $temp_snap ]; then
      hadr_role=$(head -1 $temp_snap)
      hadr_state=$(tail -1 $temp_snap)
   fi
   rm -f $temp_snap
   logger -i -p notice -t $0 "$DB2HADRDBNAME ($hadr_role, $hadr_state)"

   # hadr_role & hadr_state should now be set
   if [[ $hadr_role == "Primary" ]]; then
      if [[ $hadr_state == "Peer" ]]; then
         # Online
         rgmbrreq -o Unlock IBM.Application:$DATA_SVC_R_NAME 2> /dev/null
              logger -i -p debug -t $0 "$DB2HADRDBNAME ($hadr_role, $hadr_state).
UnLocking $DATA_SVC_R_NAME"
         rc=1
      else
```

```
         # Primary non-Peer
              logger -i -p notice -t $0 "$DB2HADRDBNAME ($hadr_role, $hadr_state).
Locking $DATA_SVC_R_NAME"
         rgmbrreq -o Lock IBM.Application:$DATA_SVC_R_NAME 2> /dev/null
         if [[ "$monitor_as_standby" == "N" ]]; then

            # Return online
            rc=1

         else
            # Return Primary Non Peer distinctly
            rc=3
         fi
      fi

   elif [[ $hadr_role == "Standby" ]]; then
      if [[ $hadr_state == "Peer" ]]; then
         rc=2
      else
         # Must return Standby states as offline to TSA
         # Only if script called directly with S option
         #  is the standby states returned distinctly
         if [[ "$monitor_as_standby" == "N" ]]; then
            # Standby return as Offline
            rc=2
         else
            rc=40
         fi
      fi
   else
      # Unknown state
      if [[ "$monitor_as_standby" == "N" ]]; then

         lockreq=$(/usr/bin/samdiag -x votes IBM.Application:${DATA_SVC_R_NAME?} 2>
/dev/null | grep -c "Type: lock")
         if [[ $lockreq -eq 0 ]]; then
            logger -i -p notice -t $0 "$DATA_SVC_R_NAME state is not known, returning
Unknown: $lockreq"

            activate=$(db2pd.exe -hadr -db ${DB2HADRDBNAME?} \
            2> /dev/null | grep -c "not activated" )

            if [[ $activate -eq 1 ]]; then

               logger -i -p notice -t $0 "DB is not activated, activating"
               db2clpex.exe -c DB2 -z log.txt -tv "activate db ${DB2HADRDBNAME}"
               sleep 30
               rc=4

            else

               rc=0
            fi

         else
            logger -i -p notice -t $0 "$DATA_SVC_R_NAME state is not known,
$DATA_SVC_R_NAME is locked, returing Offline: $lockreq"
            rc=2
         fi
      else
         rc=50
      fi
   fi
```

```
   return $rc
}

############################################################
# probehadr()
#
# Probe status of HADR on the active resource on this node
#
############################################################
probehadr()
{
if [[ "$VERBOSE" == "verbose" ]]; then
   typeset -ft $(typeset +f)
   set -x
fi

   unset instance_to_monitor
   set_candidate_P_instance
   if [[ -z "$candidate_P_instance" ]]; then
      # If we cannot find instance to monitor, return Unknown
      logger -i -p err -t $0 "Cannot find instance name!"
      exit 0
   fi
   instance_to_monitor=$candidate_P_instance
   probehadr_on_this_node
}

########################################################
# main()
########################################################
main()
{
if [[ "$VERBOSE" == "verbose" ]]; then
   typeset -ft $(typeset +f)
   set -x
fi

   probehadr
   if [ -f /tmp/.virtual_offline_${CLUSTER_APP_LABEL?} ]; then
      # Offline
      rc=2
      logger -i -p info -t $0 "'CLUSTER_APP_LABEL' ${CLUSTER_APP_LABEL?}"
   fi

   logger -i -p info -t $0 "$DB2HADRDBNAME ($hadr_role, $hadr_state). Returning $rc"
   return $rc
}



main "${@:-}"

echo $rc
exit $rc
```

$ cat hadr_start.ksh

```
#!/bin/ksh -p
#-------------------------------------------------------------------------
# (C) COPYRIGHT International Business Machines Corp. 2001-2010
# All Rights Reserved
#
# US Government Users Restricted Rights - Use, duplication or
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
#
# NAME: hadr_start.ksh
#
# (%W%) %E%  %U%
#
# FUNCTION: Start DB2 UDB, for specified RESOURCE_NAME and RESOURCE_GROUP
#
# INPUT:  hadr_start.ksh db2instp db2insts db2hadrdb verbose [S]
#
# OUTPUT: 0 if started ok
#
#-------------------------------------------------------------------------
PATH=/bin:/usr/bin:/sbin:/usr/sbin:/usr/sbin/rsct/bin:"/dev/fs/C/Program
Files/IBM/SQLLIB/BIN":/dev/fs/C/WINDOWS/system32
export PATH=$PATH:.
logger -i -p info -t $0 "$*"

RESOURCE_METHOD=start

PROGNAME=$(basename $0)
PROGPATH=$(dirname $0)
DB2HADRINSTANCE1=${1?}
DB2HADRINSTANCE2=${2?}
DB2HADRDBNAME=${3?}
VERBOSE=${4:-verbose}
start_as_standby=${5:-N}

export CT_MANAGEMENT_SCOPE=2
export DB2INSTANCE=${DB2HADRINSTANCE1?}

if [[ "$VERBOSE" == "verbose" ]]; then
   typeset -ft $(typeset +f)
   set -x
else
   # Close stdout and stderr
   exec 2> /dev/null
   exec 1> /dev/null
   set +x
fi

PROBE=
START=$PROGNAME
STOP=
RT_BASEDIR=${PROGPATH?}
SVC_PROBE=${RT_BASEDIR}/${PROBE:-hadr_monitor.ksh}
START_CMD=$RT_BASEDIR/${START:-hadr_start.ksh}
STOP_CMD=$RT_BASEDIR/${STOP:-hadr_stop.ksh}

if [[ $DB2HADRINSTANCE1 != $DB2HADRINSTANCE2 ]]; then
   RGNAME1=db2_${DB2HADRINSTANCE1?}_
   RSNAME1=db2_${DB2HADRINSTANCE1?}_
   RGNAME2=db2_${DB2HADRINSTANCE2?}_
   RSNAME2=db2_${DB2HADRINSTANCE2?}_

   Resource=${RSNAME1?}
```

```
   ResourceClass=IBM.Application
   NodeRG1=$(lsrsrc-api -s ${ResourceClass?}::'Name like
"'${Resource?}%'"'::NodeNameList | tr "{" " " | tr "}" " " | tr "." " " | awk '{print
$1}' | tail -1)

   Resource=${RSNAME2?}
   ResourceClass=IBM.Application
   NodeRG2=$(lsrsrc-api -s ${ResourceClass?}::'Name like
"'${Resource?}%'"'::NodeNameList | tr "{" " " | tr "}" " " | tr "." " " | awk '{print
$1}' | tail -1)

else
   Resource=db2_${DB2HADRINSTANCE1?}_
   ResourceClass=IBM.Application

   NodeRG1=$(lsrsrc-api -s ${ResourceClass?}::'Name like
"'${Resource?}%'"'::NodeNameList  | tr "{" " " | tr "}" " " | tr "." " " | awk '{print
$1}' | sort -n | uniq | head -1)
   NodeRG2=$(lsrsrc-api -s ${ResourceClass?}::'Name like
"'${Resource?}%'"'::NodeNameList  | tr "{" " " | tr "}" " " | tr "." " " | awk '{print
$1}' | sort -n | uniq | tail -1)

   RGNAME1=db2_${DB2HADRINSTANCE1?}_${NodeRG1?}_0-rg
   RSNAME1=db2_${DB2HADRINSTANCE1?}_${NodeRG1?}_0-rs
   RGNAME2=db2_${DB2HADRINSTANCE2?}_${NodeRG2?}_0-rg
   RSNAME2=db2_${DB2HADRINSTANCE2?}_${NodeRG2?}_0-rs
fi

CLUSTER_APP_LABEL=db2_${DB2HADRINSTANCE1}_${DB2HADRINSTANCE2}_${DB2HADRDBNAME?}

# Name the resource group and the resource
DATA_SVC_RG_NAME=${CLUSTER_APP_LABEL}-rg
DATA_SVC_R_NAME=${CLUSTER_APP_LABEL}-rs


############################################################
# error_exit
############################################################
error_exit()
{
   typeset exit_code="${rc:-1}"
   logger -i  -p err -t $0 "exiting with $rc"
   exit $rc
}

############################################################
# resource_group_state_on_this_node()
#
# INPUT: ResourceClass, Resource, ResourceType, Node
# OUTPUT:rg_state [Online=1 || Offline]
#
############################################################
resource_group_state_on_this_node()
{
   lsrsrc-api -s ${ResourceClass}::'Name="'${Resource}'" '"${ResourceType}"'
'::NodeNameList

}

############################################################
# HADR_partner_node_state()
#
# INPUT: HADR_REMOTE_HOST [hostname]
# OUTPUT:remote_node_alive [Online || Offline]
```

```
#
############################################################
HADR_partner_node_state()
{
   remote_node_alive=Online

   if [[ ! -z $HADR_REMOTE_HOST ]]; then
      remote_host_in_tsa_format=$(echo ${HADR_REMOTE_HOST?} | tr "." " " | awk '{print
$1}')
      remote_node_alive=$(lsrpnode -x  | sort | awk '{print $1 " " $2}' | grep
${remote_host_in_tsa_format?} | awk '{print $2}')
   else
      # Is any node in domain offline ...
      nno=$(lsrpnode -x -O | wc -l | awk '{print $1}')
      if [[ $nno == 0 ]]; then
         remote_node_alive=Online
      else
         remote_node_alive=Offline
      fi
   fi
}

############################################################
# set_candidate_P_instance()
#
# INPUT: RSNAME1, RSNAME2, DB2HADRINSTANCE1, DB2HADRINTANCE2
# OUTPUT:candidate_P_instance
#
############################################################
set_candidate_P_instance()
{
   Node=$(hostname | tr "." " " | awk '{print $1}')

   if   [[ $NodeRG1 == $Node && $NodeRG2 != $Node ]]; then
      candidate_P_instance=$DB2HADRINSTANCE1
      HADR_REMOTE_HOST=$NodeRG2
      forceRGOfflineInCaseOfByForce=$RGNAME2

   elif [[ $NodeRG1 != $Node && $NodeRG2 == $Node ]]; then
      candidate_P_instance=$DB2HADRINSTANCE2
      HADR_REMOTE_HOST=$NodeRG1
      forceRGOfflineInCaseOfByForce=$RGNAME1
   else
      # either both instances are up in resource groups
      # on this node, or neither is
      # figure out the one that wants to be primary

      rm -rf log.txt
       db2clpex.exe -c DB2 -z log.txt -tv "get db cfg for ${DB2HADRDBNAME?}"
       HADR_LOCAL_HOST1=`cat log.txt | grep HADR_LOCAL_HOST | awk '{print $7}'`
       HADR_LOCAL_HOST2=`cat log.txt | grep HADR_LOCAL_HOST | awk '{print $7}'`
           rm -rf log.txt

      if [[ $HADR_LOCAL_HOST1 == $Node ]]; then
         candidate_P_instance=$DB2HADRINSTANCE1
         forceRGOfflineInCaseOfByForce=$RGNAME2
         HADR_REMOTE_HOST=$NodeRG2
      elif [[ $HADR_LOCAL_HOST2 == $Node ]]; then
         candidate_P_instance=$DB2HADRINSTANCE2
         forceRGOfflineInCaseOfByForce=$RGNAME1
         HADR_REMOTE_HOST=$NodeRG1
      else
         candidate_P_instance=
         forceRGOfflineInCaseOfByForce=
```

```
         HADR_REMOTE_HOST=
      fi
   fi
}

#############################################################
# starthadr()
#############################################################
starthadr()
{

   set_candidate_P_instance
   instance_to_start=${candidate_P_instance}

   $SVC_PROBE ${DB2HADRINSTANCE1?} ${DB2HADRINSTANCE2?} ${DB2HADRDBNAME?} ${VERBOSE?} S
   rc=$?

   HADR_partner_node_state

   if [[ $remote_node_alive == "Online" ]]; then
      # Bring up HADR as Primary on this node
      if [ $rc -eq 1 ]; then
         # already primary
         rc=0
      elif [ $rc -eq 2 ]; then
         # currently standby,peer
         # takeover (no force)
         logger -i  -p notice -t $0 "db2 takeover hadr on db ${DB2HADRDBNAME?}
${SVC_PROBE}"
         db2clpex.exe -c DB2 -z log.txt -tv "takeover hadr on db ${DB2HADRDBNAME?}"
              logger -i  -p notice -t $0 "Wait 20 seconds after hadr takeover command
issued..."
         sleep 20

               $SVC_PROBE ${DB2HADRINSTANCE1?} ${DB2HADRINSTANCE2?} ${DB2HADRDBNAME?}
${VERBOSE?}
         rc1=$?
         if [ $rc1 -ne 1 ]; then
                  logger -i  -p err -t $0 "TAKEOVER FAILED : $rc1"
         fi

      elif [ $rc -eq 40 ]; then
         :
         logger -i  -p err -t $0 "*** Database ${DB2HADRDBNAME} is not in Peer State,
old Primary still Online"
      fi # Bring up HADR as Primary on this machine

   else
      # Old primary machine is offline
      if [ $rc -eq 2 ]; then
         # Standby is currently in Peer State
         #
         # To bring up standby, will now do a TAKEOVER BY FORCE
         # No need to block until resource group is offline, we have verified
         # that the node is down already
         # To bring up standby, will now do a TAKEOVER BY FORCE
         :
         logger -i  -p notice -t $0 "db2 takeover hadr on db ${DB2HADRDBNAME?} by
force"
         db2clpex.exe -c DB2 -z log.txt -tv "takeover hadr on db ${DB2HADRDBNAME?} by
force"
         logger -i  -p notice -t $0 "NOTICE: Takeover by force issued"
      elif [ $rc -eq 40 ]; then
         # Standby is currently not in Peer State
```

```
         :
         logger -i  -p err -t $0 "*** Database ${DB2HADRDBNAME} is not in Peer State,
old Primary Offline"
         logger -i  -p notice -t $0 "db2 takeover hadr on db ${DB2HADRDBNAME?} by
force"
         db2clpex.exe -c DB2 -z log.txt -tv "takeover hadr on db ${DB2HADRDBNAME?} by
force"
         logger -i  -p notice -t $0 "NOTICE: Takeover by force issued"
      fi
   fi # Bring up HADR on this machine

   # Return state
   sleep 30

   $SVC_PROBE ${DB2HADRINSTANCE1?} ${DB2HADRINSTANCE2?} ${DB2HADRDBNAME?} ${VERBOSE?} S
   rcs=$?
   logger -i  -p notice -t $0 "On standby node after Probe S - rcs= ${rcs}"

   # Online succesful must return 0 whilst monitor returns 1
   # for Primary in Peer State and 3 for Primary not Peer
   if [ $rcs -eq 1 ]; then
      rc=0
   elif [ $rcs -eq 3 ]; then
      rc=0
      # Anything else, map directly from monitor
   else
      rc=$rcs
   fi

   return $rc
}

#########################################################
# main()
#########################################################
main()
{
   rm -f /tmp/.virtual_offline_${CLUSTER_APP_LABEL?}
   starthadr
   logger -i -p info -t $0 "$CLUSTER_APP_LABEL returning $rc"
   return $rc
}

#
if [[ "$VERBOSE" == "verbose" ]]; then
   typeset -ft $(typeset +f)
fi

main"${@:-}"

echo $rc
exit $rc
```

$ cat hadr_stop.ksh

```
#!/bin/ksh -p
#-------------------------------------------------------------------------
# (C) COPYRIGHT International Business Machines Corp. 2001-2010
# All Rights Reserved
#
# US Government Users Restricted Rights - Use, duplication or
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
#
# NAME: hadr_stop.ksh
#
# (%W%) %E%  %U%
#
# FUNCTION: Stop HADR for specified HADR pair
#
# INPUT:  hadr_stop.ksh db2instp db2insts db2hadrdb verbose [S]
#
# OUTPUT: 0 if service stopped
#
#-------------------------------------------------------------------------
PATH=/bin:/usr/bin:/sbin:/usr/sbin:/usr/sbin/rsct/bin:"/dev/fs/C/Program
Files/IBM/SQLLIB/BIN"
export PATH=$PATH:.:PATH

logger -i  -p info -t $0 "$*"

RESOURCE_METHOD=stop

PROGNAME=$(basename $0)
PROGPATH=$(dirname $0)
RT_BASEDIR=$PROGPATH
DB2HADRINSTANCE1=${1?}
DB2HADRINSTANCE2=${2?}
DB2HADRDBNAME=${3?}
VERBOSE=${4:-verbose}
stop_standby_also=${5:-S}
VIRTUAL_STOP_ONLY="yes"
PROBE=
START=
STOP=$PROGNAME
SVC_PROBE=${RT_BASEDIR}/${PROBE:-hadr_monitor.ksh}
START_CMD=$RT_BASEDIR/${START:-hadr_start.ksh}
STOP_CMD=$RT_BASEDIR/${STOP:-hadr_stop.ksh}

export CT_MANAGEMENT_SCOPE=2
export DB2INSTANCE=${DB2HADRINSTANCE1?}

CLUSTER_APP_LABEL=db2_${DB2HADRINSTANCE1?}_${DB2HADRINSTANCE2?}_${DB2HADRDBNAME?}

# Name the resource group and the resource
DATA_SVC_RG_NAME=${CLUSTER_APP_LABEL}-rg
DATA_SVC_R_NAME=${CLUSTER_APP_LABEL}-rs


if [[ "$VERBOSE" == "verbose" ]]; then
   typeset -ft $(typeset +f)
   set -x
else
   # Close stdout and stderr
   exec 2> /dev/null
   exec 1> /dev/null
   set +x
fi
```

```
#############################################################
# error_exit
#############################################################
error_exit()
{
   typeset exit_code="${rc:-1}"
   logger -i  -p err -t $0 " exiting with $exit_code"
   exit $exit_code
}


#############################################################
# resource_group_state_on_this_node()
#
# INPUT: ResourceClass, Resource, ResourceType, Node
# OUTPUT:rg_state [Online=1 || Offline]
#
#############################################################
resource_group_state_on_this_node()
{
   OpState=$( lsrsrc-api -s ${ResourceClass}::'Name="'${Resource}'" '"${ResourceType}"'
&& NodeNameList={"'${Node}'"} '::OpState )

   rg_state=${OpState?}
}

#############################################################
# set_candidate_P_instance()
#
# INPUT: RSNAME1, RSNAME2, DB2HADRINSTANCE1, DB2HADRINTANCE2
# OUTPUT:candidate_P_instance
#
#############################################################
set_candidate_P_instance()
{
set -x
   Node=$(hostname | tr "." " " | awk '{print $1}')

   if   [[ $NodeRG1 == $Node && $NodeRG2 != $Node ]]; then
      candidate_P_instance=$DB2HADRINSTANCE1
      forceRGOffline=$RGNAME1
   elif [[ $NodeRG1 != $Node && $NodeRG2 == $Node ]]; then
      candidate_P_instance=$DB2HADRINSTANCE2
      forceRGOffline=$RGNAME2
   else
      # either both instances are up in resource groups
      # on this node, or neither is
      # figure out the one that wants to be primary

      rm -rf log.txt
      db2clpex.exe -c DB2 -z log.txt -tv "get db cfg for ${DB2HADRDBNAME?}"
      HADR_LOCAL_HOST1=cat log.txt | grep HADR_LOCAL_HOST | awk '{print $7}'
      HADR_LOCAL_HOST2=cat log.txt | grep HADR_LOCAL_HOST | awk '{print $7}'
      rm -rf log.txt

      if [[ $HADR_LOCAL_HOST1 == $Node ]]; then
         candidate_P_instance=$DB2HADRINSTANCE1
         forceRGOffline=$RGNAME1
      elif [[ $HADR_LOCAL_HOST2 == $Node ]]; then
         candidate_P_instance=$DB2HADRINSTANCE2
         forceRGOffline=$RGNAME2
      else
         candidate_P_instance=
```

```
         forceRGOfflineInCaseOfByForce=
      fi
   fi
}

#############################################################
# stophadr()
#
# Stop HADR on instance that is running local to this node
#############################################################
stophadr()
{
   set_candidate_P_instance
   instance_to_stop=$candidate_P_instance

   if [[ ! -z "$instance_to_stop" ]]; then
      chrg -o Offline ${forceRGOffline?}
      db2_kill.bat
   fi

   rc=0
   return $rc
}

#########################################################
# main()
#########################################################
main()
{
   touch /tmp/.virtual_offline_${CLUSTER_APP_LABEL?}
   rc=$?

   if [[ "$VIRTUAL_STOP_ONLY" != "yes" ]]; then
      stophadr  || error_exit $?
   fi
   logger -i -p info -t $0 "$CLUSTER_APP_LABEL returning $rc"
   return $rc
}
#
if  [[  "$VERBOSE" == "verbose"   ]]; then
   typeset -ft $(typeset   +f)
fi

main"${@:-}"

echo    $rc
exit    $rc
```

Appendix D: Automated HADR Reintegration

As of DB2 Version 9.7 Fix Pack 8, automatic HADR reintegration is supported. This feature allows the HADR pair to automatically regain peer state upon an outage of either the standby or primary server. The minimum operating system and software requirements are as follows:

- Operating system: Windows Server 2008 R2 SP1 with Microsoft Hotfix KB2639164
- Tivoli product: IBM Tivoli System Automation for Multiplatforms Version 3.2.2.3

If the HADR primary and standby servers suffer a concurrent outage, you might have to perform manual recovery to regain HADR peer state.

Manually recovering from a concurrent primary and standby server outage

If the HADR pair fails to automatically recover from a situation where the primary and standby servers suffered a concurrent outage, the lssam command output is similar to what follows:

```
$ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs Request=Lock
               |- Offline IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock1
               '- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip
               |- Offline IBM.ServiceIP:db2ip:spock1
               '- Online IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

To recover from this situation, issue the following command from standby server:

```
db2 start hadr on db hadrdb as standby
```

Recovering from a "Failed offline" state

If you implemented automatic HADR reintegration, the standby server's HADR resource might appear as `Failed offline` in the lssam command output after an automated failover takes place, as shown in the following example:

```
$ lssam
Online IBM.ResourceGroup:db2_db2inst_db2inst_HADRDB-rg
Control=MemberInProblemState Nominal=Online
      |- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs
      Control=MemberInProblemState
                |- Failed offline IBM.Application:db2_db2inst_db2inst_HADRDB-
                rs:spock1
                '- Online IBM.Application:db2_db2inst_db2inst_HADRDB-rs:spock2
      '- Online IBM.ServiceIP:db2ip
                |- Offline IBM.ServiceIP:db2ip:spock1
                '- Online IBM.ServiceIP:db2ip:spock2
Online IBM.ResourceGroup:db2_db2inst_spock1_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock1_0-rs:spock1
Online IBM.ResourceGroup:db2_db2inst_spock2_0-rg Nominal=Online
      '- Online IBM.Application:db2_db2inst_spock2_0-rs:spock2
Online IBM.Equivalency:virpubnic_spock1_spock2
      |- Online IBM.NetworkInterface:B13D316B-A16B-4D34-84D3-10B0D743436B:spock1
      '- Online IBM.NetworkInterface:78B5ACFD-0B8A-4AD8-9C6B-43B651541F1F:spock2
```

To recover from this situation, see “The resetrsrc command - A brief how-to guide” (**http://www.ibm.com/support/docview.wss?uid=swg21458938**).

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada