IIB and MQ HA on Pure Applications

IIB is locally bound to Queue Managers

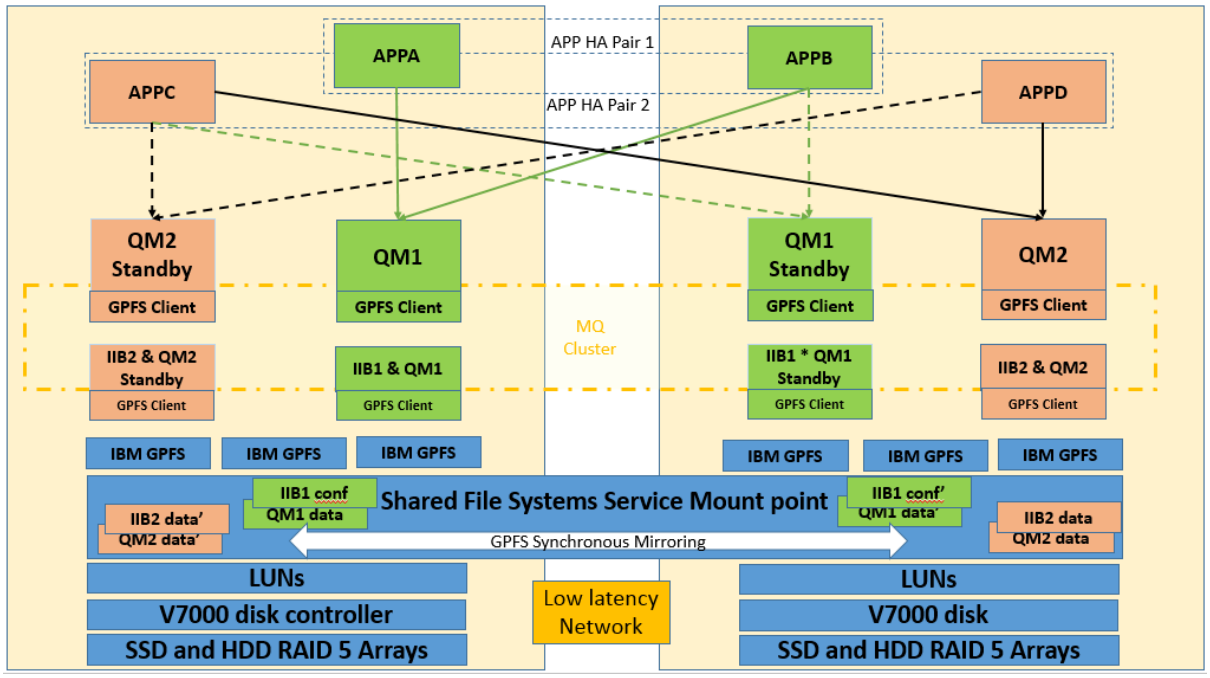
MQ and IIB logs and data on a GPFS in the pattern

Pattern Description

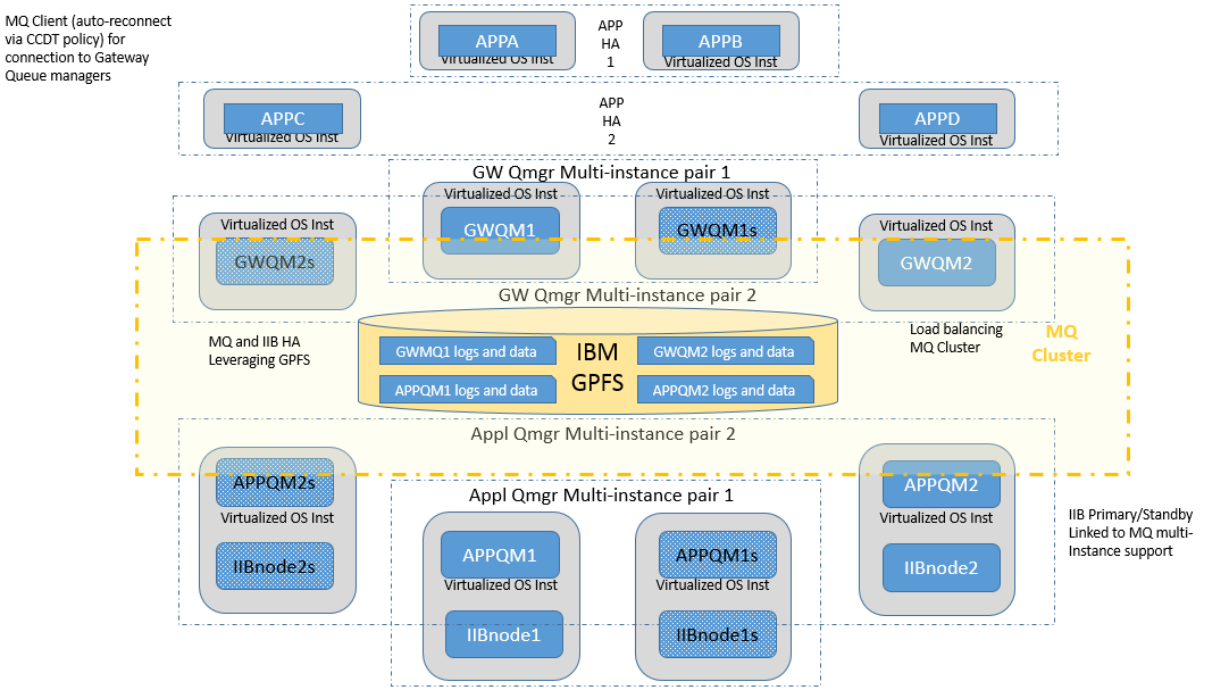
Table of Contents

Context diagrams

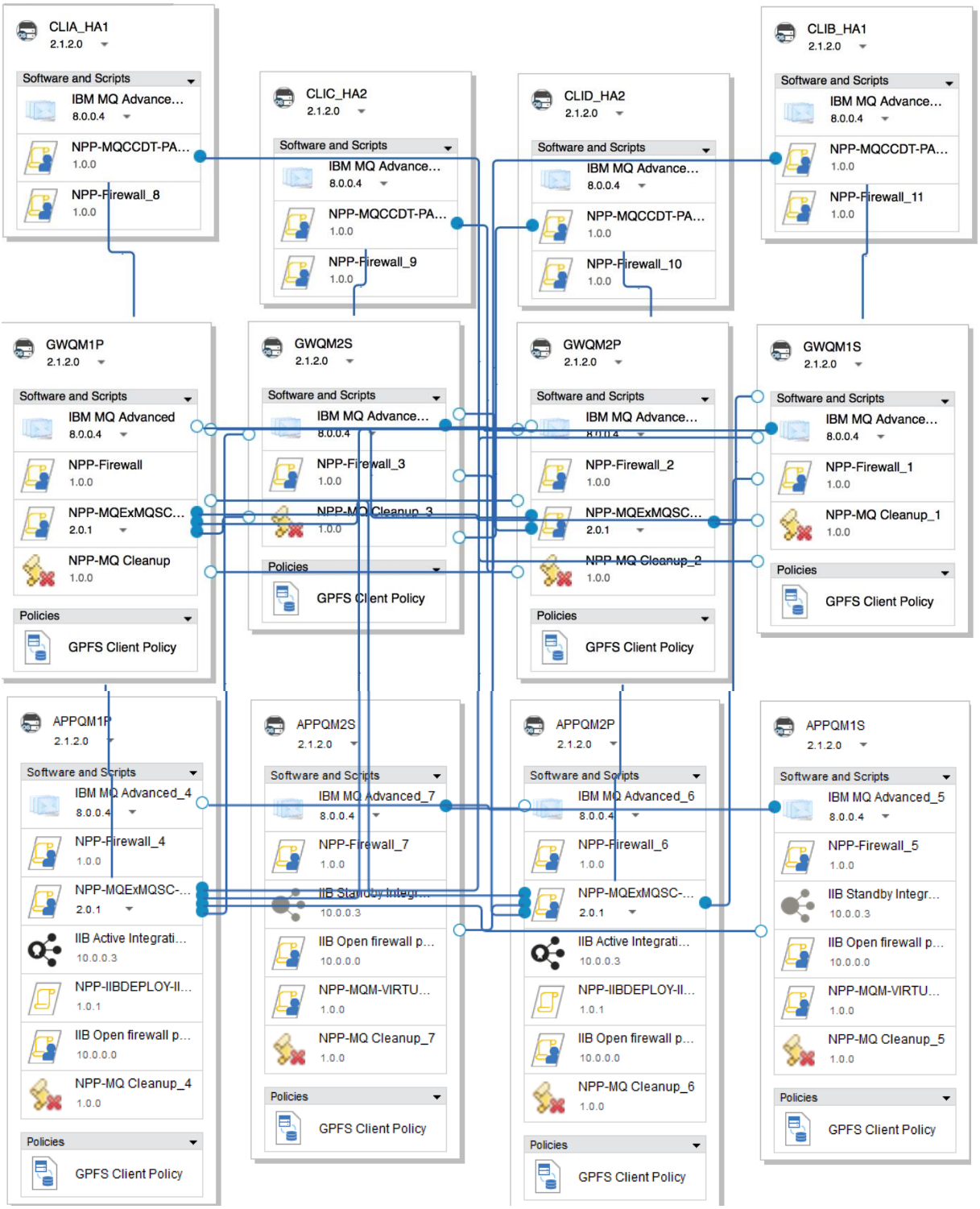
High level physical view



High level logical MQ/IIB view



Pattern Diagram- DA1 - MQ and IIB local HA on GPFS new (FULL) v2.4



Pattern Definition

Overview Description

The “DA1 - MQ and IIB local HA on GPFS” v2.4 features

MQ Cluster

- 2 nodes wide
    - 2 Gateway queue managers
    - 2 Application queue managers
- Gateway and Application queue managers
- Non-dedicated full cluster repositories on Gateway queue managers
- Gateway to Application queue manager inbound load balancing via RQ (receipt queues)
- Application to Gateway queue manager outbound load balancing via EQ (emission queues)
- Cluster queues defined with DEFBIND(NOTFIXED)

MQ multi-instance HA pairs

- Each queue manager has a single Multi-instance standby partner

IBM Integration Bus

- 2 IIB Nodes
- Each IIB Node is associated with an Application Queue Manager (Local binding)
- IIB Multi-instance standby nodes are deployed for each IIB node.
- “Loop back” message flows are deployed to
    - MQGet from RQs
    - MQPut to EQs

MQ Client Application Gateway nodes

- 4 MQ Client nodes
- MQ Installation only no queue managers
- MQ CCDT used to resolve a Gateway queue manager per pair
    - MQ client HA1 pair resolve GWQM1 primary and standby
    - MQ client HA2 pair resolve GWQM2 primary and standby

Pattern Parameters

Passwords

Virtuser = passw0rd

Root = passw0rd

* **Name**

ASL - NPP MQ and IIB local HA on GPFS new

* **Version**

2.4

Description

based NPP MQ and IIB HA on GPFS v2.3
add a PAG layer with MQ S/W installed and
CCDT table creation

* **Type**

- (•) Pattern
- ( ) Pattern Template

Lock option for plug-in usage

- (•) Unlock plug-ins
- ( ) Lock all plug-ins except Foundation plug-ins
- ( ) Lock all plug-ins

▾ Pattern-level Parameters

Add new parameter

* **Password (root)**

••••••••

••••••••

* **Password (virtuser)**

••••••••

••••••••

Pattern Nodes

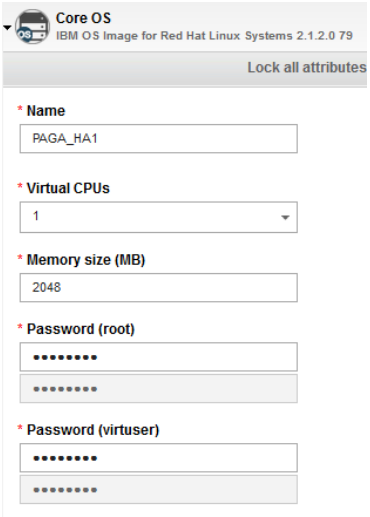
MQ Client Application Gateway nodes

Description

Place holder virtual machine set up to house MQ Client connected applications. In this pattern the MQ Client application will work in HA pairs. Each pair resolving connectivity to a Gateway Queue Manager Multi-instance HA pair via the Client Channel Definition Table (CCDT)

MQ Client Application Gateway Parameters

HA1-A



HA1-B

As above for HA1-A

HA2-C

As above for HA1-A

HA2-D

As above for HA1-A

MQ Gateway nodes

Description

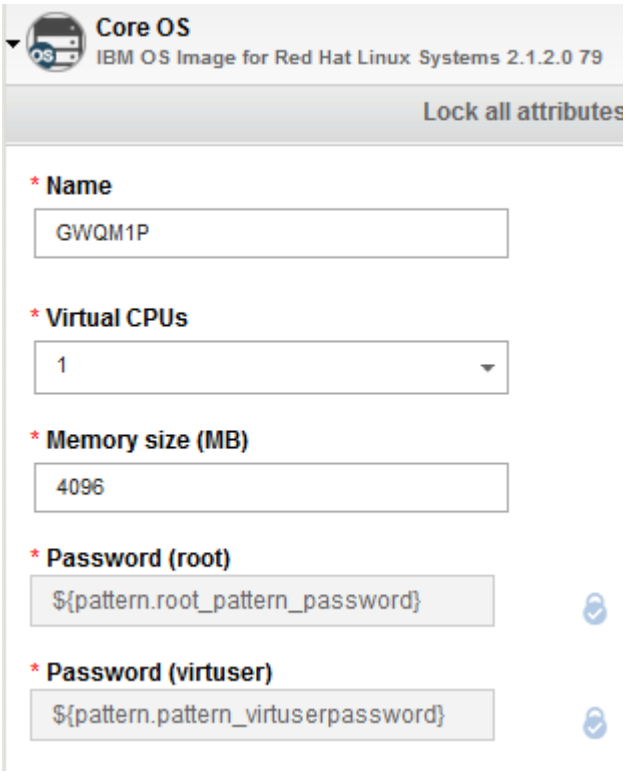
Twin MQ Gateway queue managers each with a multi-instance partner working in an MQ cluster with the MQ Application queue managers. Gateway queue managers host a full cluster repository each.

EQs are local and shared in the cluster.

MQ Security is disabled.

MQ Gateway Parameters

GWQM1P – primary node 1



GWQM1S – standby node 1

As above for GWQM1P

GWQM2P – primary node 2

As above for GWQM1P

GWQM2S – standby node 2

As above for GWQM1P

MQ Application/IIB nodes

Description

Twin MQ Application queue managers each with a multi-instance partner working in an MQ cluster with the MQ Gateway queue managers.

Primary Application queue managers “host” a primary IIB node.

Standby Application queue managers “host” a standby IIB node.

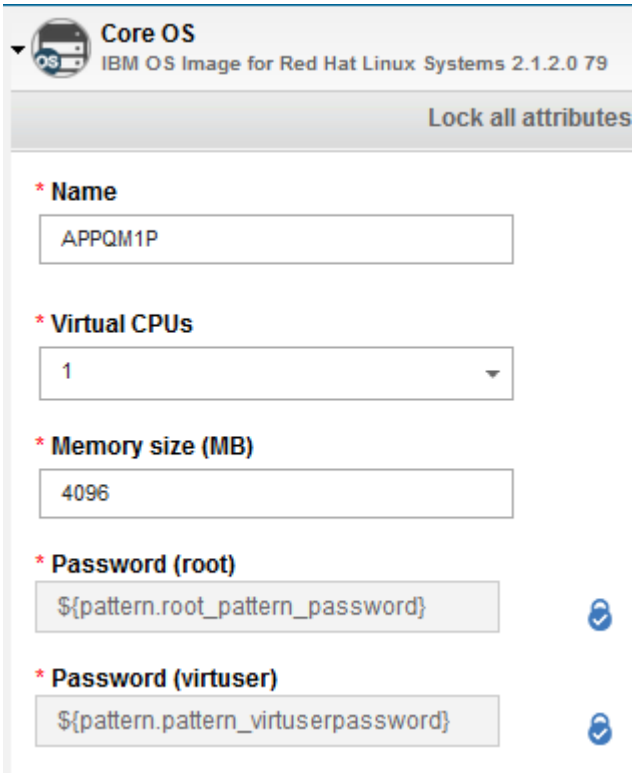
IIB has “loop back” flows deployed serving RQs and putting to EQs

RQs are local and shared in the MQ cluster.

MQ Security is disabled.

MQ Application Parameters

APPQM1P – primary node 1



APPQM1S – standby node 1

As above for APPQM1P

APPQM2P – primary node 2

As above for APPQM1P

APPQM2S – standby node 2

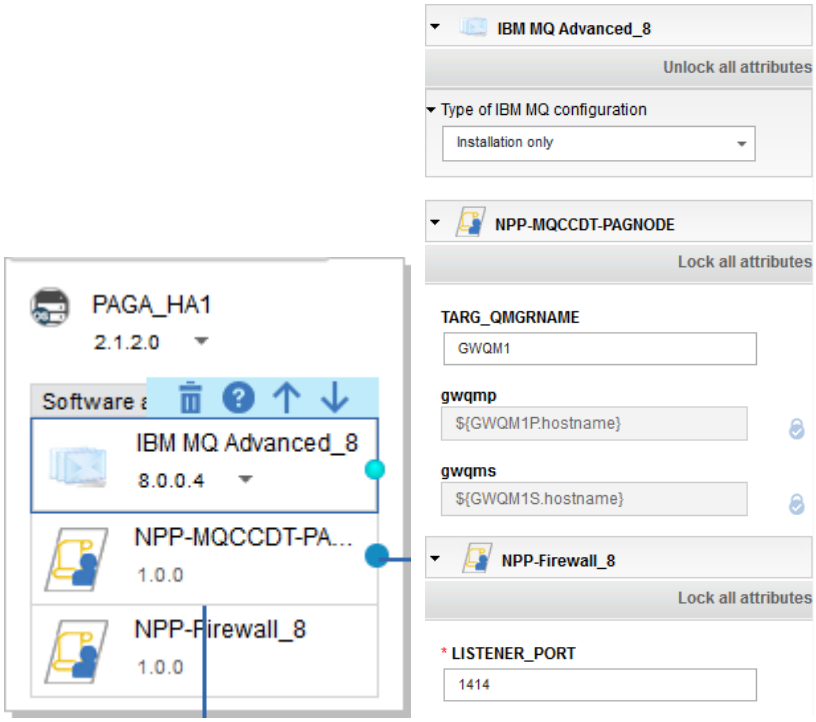
As above for APPQM1P

Pattern Software Parts

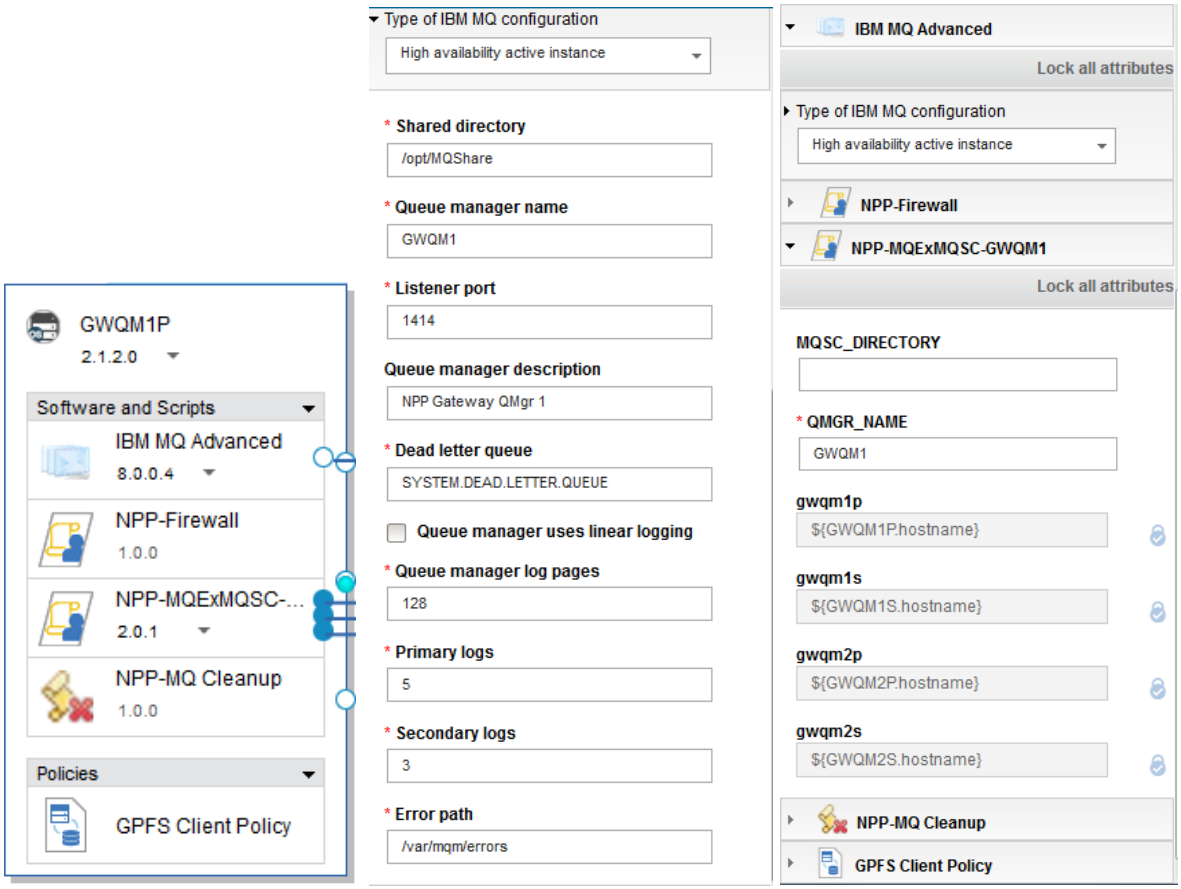
IBM MQ v8.0.0.4

IBM MQ version 8 fix pack 4

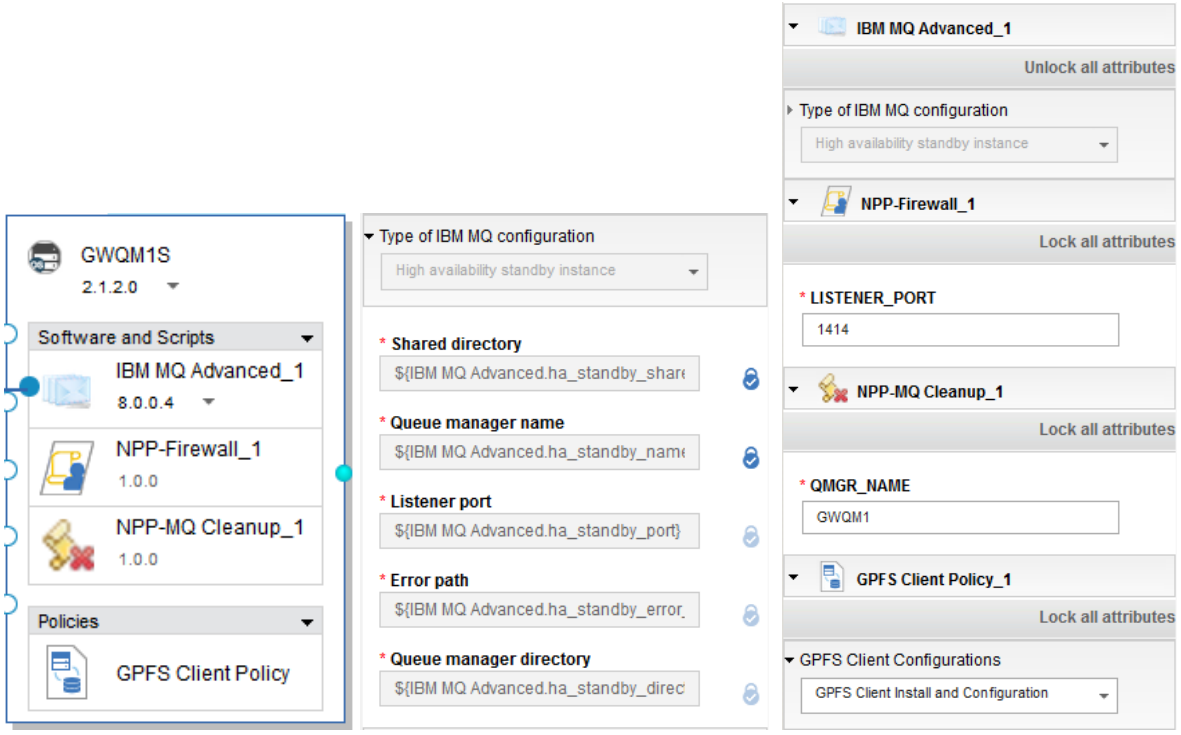
MQ Client Application Gateway Parameters



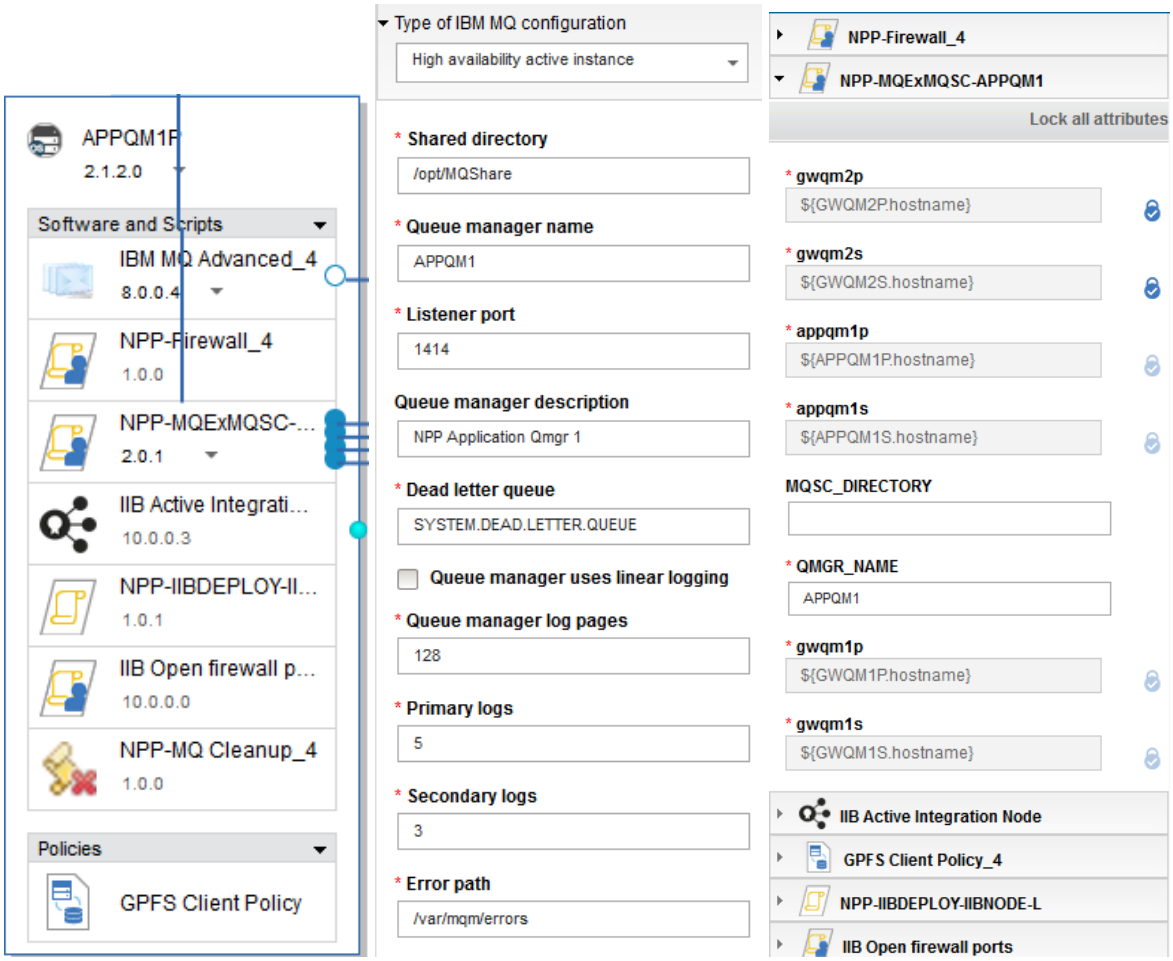
MQ Gateway Primary Parameters



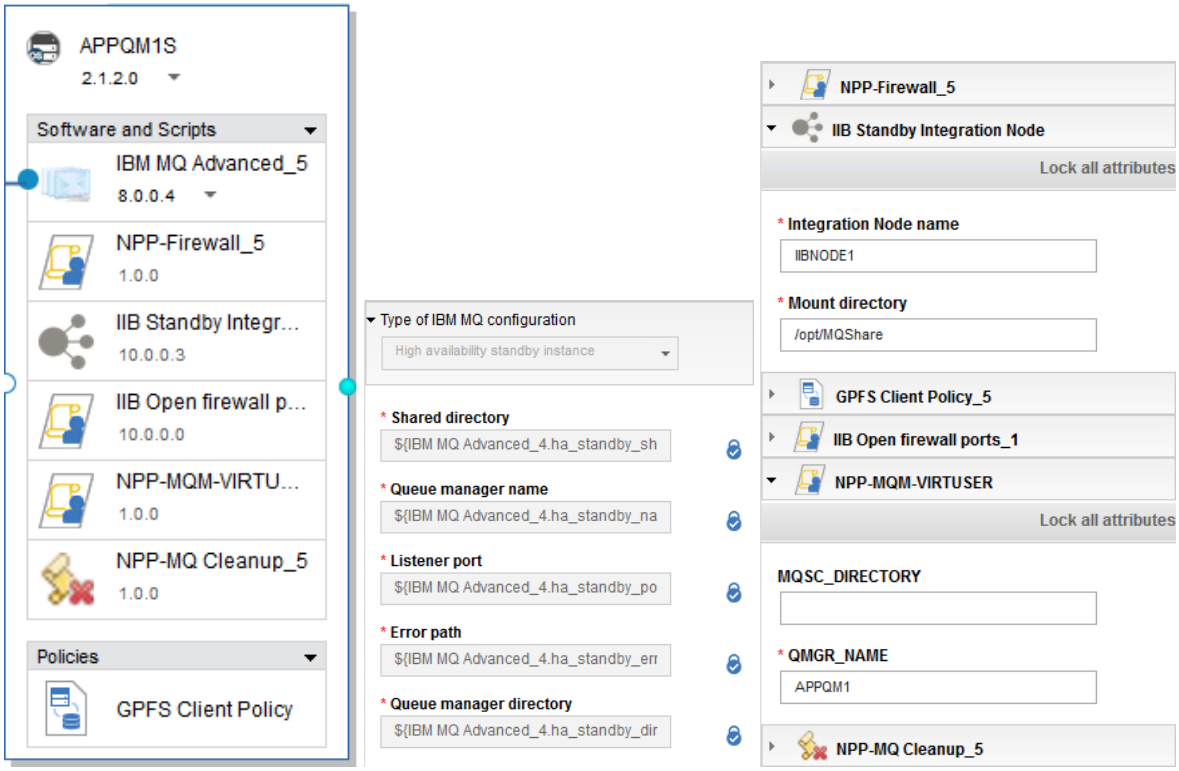
MQ Gateway Standby Parameters



MQ Application Primary Parameters



MQ Application Standby Parameters

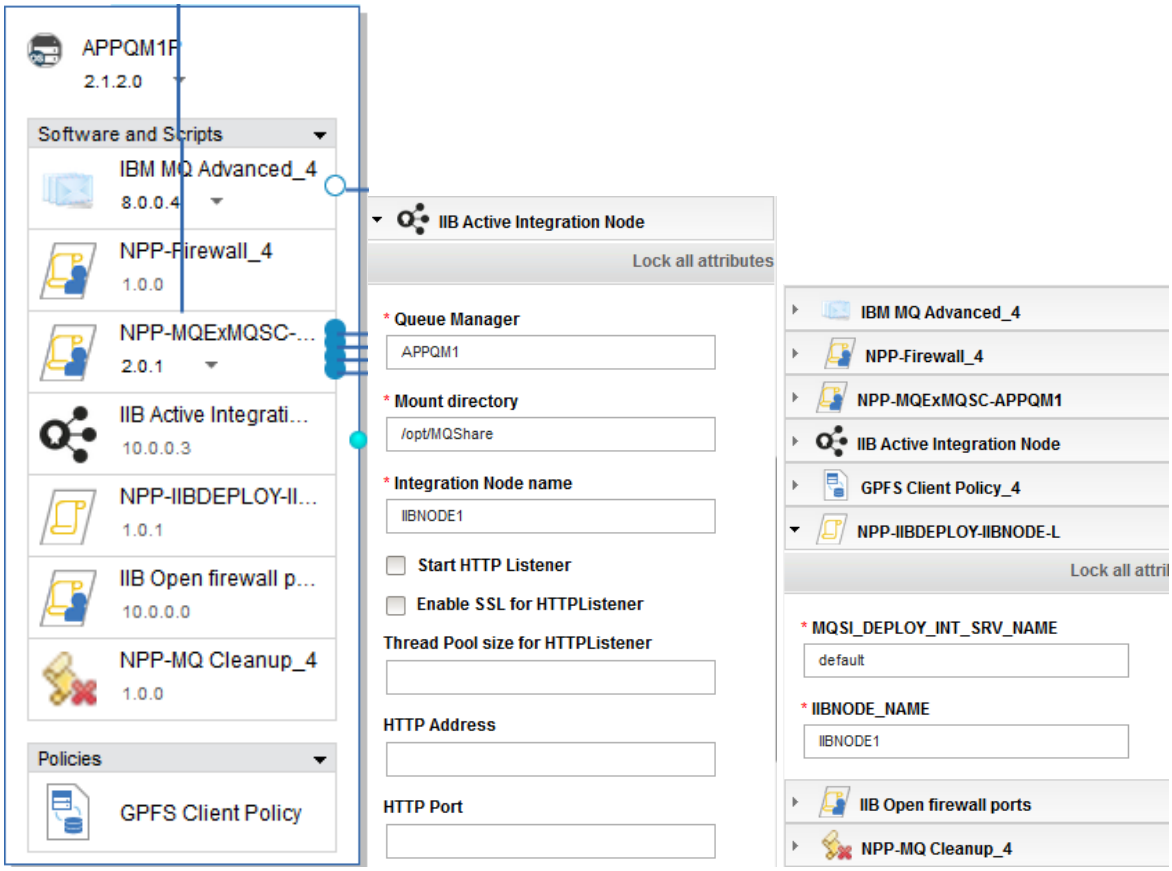


IBM Integration Bus V10.0.0.n

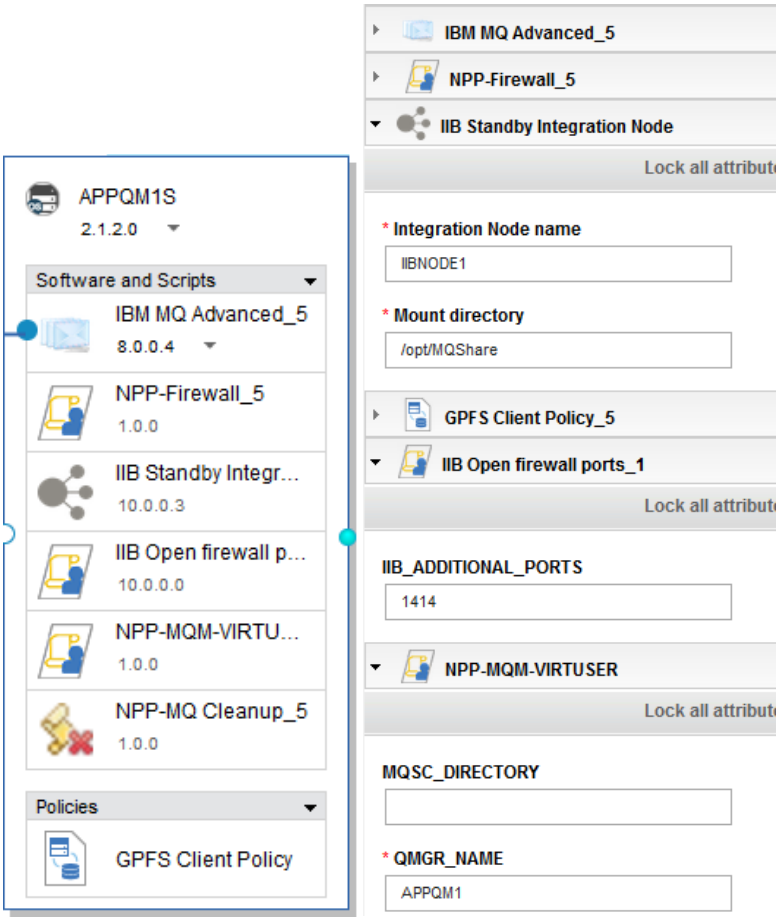
The pattern was built against IBM Integration Bus 10.0.0.3. Since delivering this version the fix pack 4 for IIB, 10.0.0.4 has been released. This fix pack resolves a problem where by the NPP-MQM-VIRTUSER script package is no longer required in the IIB Standby nodes. If you are using IIB v10.0.0.4 you may remove that script package.

The NPP-firewall and IIB Open firewall ports script packages are doing the same job in the IIB/Application nodes. Assuming no additional ports for IIB specifically are required the one or other script package could be removed in these nodes.

IIB Primary Parameters on MQ Application nodes



IIB Standby Parameters on MQ Application nodes

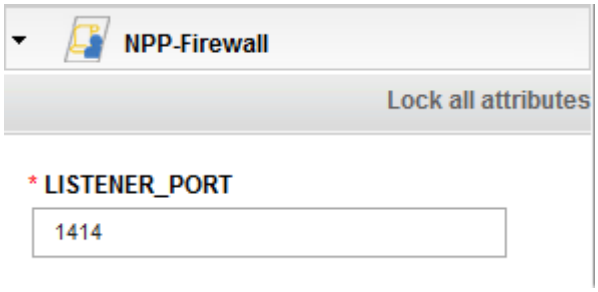


Script package descriptions

NPP-firewall

Open a MQ Listener port in the firewall.

Parameters



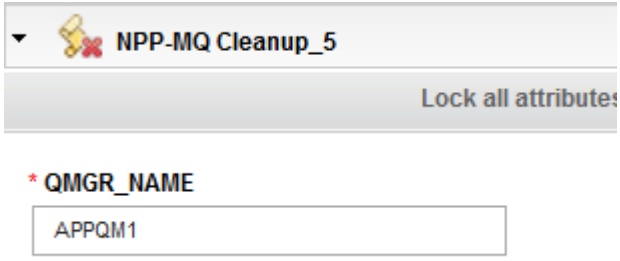
NPP-MQ Cleanup

This script package should be run when “tearing down” a pattern instance. The intention is to delete the queue managers and clean up their data log files on shared storage. I have found that this does not work very well and a manual clean up is best.

At MQ v8.0.0.3 and earlier if you do not clean up, repeated instantiation of the same pattern, which we often do when demoing and developing patterns, the Queue managers come up and pick up the old logs and data files. On the up side, the logs are replayed and you get persistent messages reloaded from previous pattern instances – good old MQ doing its job. On the down side, the exclusive file lock to determine if an HA primary is running is not honoured and both the primary and standby come up – Split brain which is bad.

At MQ v8.0.0.4 the unique pattern instance ID is inserted into the log/data file directory structure and you can no longer pick up previous iterations files. Problem solved! It is still worth going in and cleaning up from time to time.

Parameters



Source code

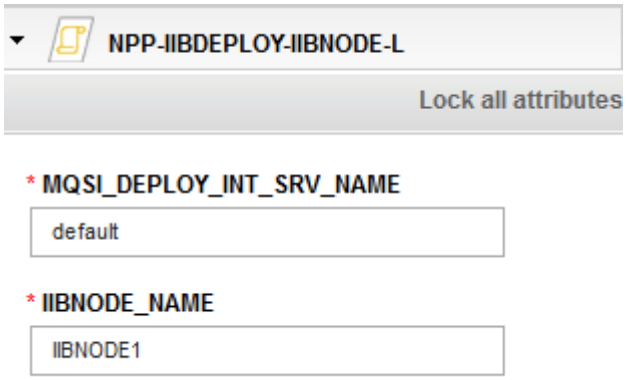
Not updated for the introduction of the pattern instance ID

```
./execute.sh "sudo su - mqm      -c \"dspmq\""

./execute.sh "sudo su - mqm -c \"endmqm -i $QMGR_NAME\""

./execute.sh "sudo su - mqm -c \"dltmqm -z $QMGR_NAME\""

./execute.sh "sudo su - mqm      -c \"dspmq\""

echo "Deleting GPFS folder."

./execute.sh "rm -rf /opt/MQShare/$QMGR_NAME"
```

NPP-IIBDEPOY-IIBNODE-L

The “-L” stands for local binding mean that the BAR contained in this script package contains a single flow to MQGet from the RQ of the queue manager associated with the IIB node and MQPut to the EQ of the same queue manager. The message flow only interacts with its associated IIB node queue manager.

Parameters



Source code

```
./execute.sh "mqsideploy $IIBNODE_NAME -e
```

NPP-MQCCDT-PAGNODE

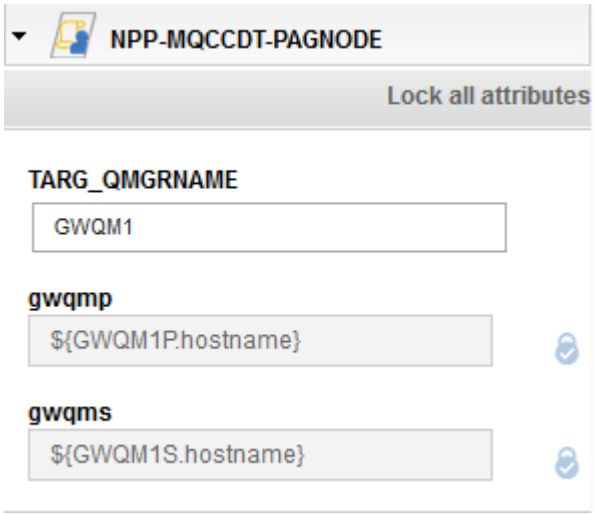
For use with MQ Client application wishing to resolve more than one MQ queue manager. In this example we have 2 queue managers that are a multi-instance pair (A primary and a standby). The script package could be extended to be greater than 2 queue managers if required.

The pure app pattern wiring is used to obtain the IP addresses of the nodes hosting the two queue managers. These are then substituted for placeholders in the script package runmqsc input file such that the CCDT is created in the node with real IP addresses. The CCDT is created in the default location with the default name such that the MQ client code picks it up.

The client code checks the MQclient.ini file, MQSERVER environment variable and MQCHLTAB environment variable none of which are set before it goes after a default location CCDT.

I have supplied but not used a script package called NPP-SET-MQENV-1.0.0.1 that can be used to set up environment variables such as MQSERVER or MQCHLTAB

Parameters



MQSC File

```
DEFINE CHANNEL(TARG_QMGRNAME) +
        CHLTYPE(CLNTCONN) +
        TRPTYPE(TCP) +
        CONNAME('gwqmp(1414),gwqms(1414)') +
        QMNAME(TARG_QMGRNAME) +
        REPLACE
```

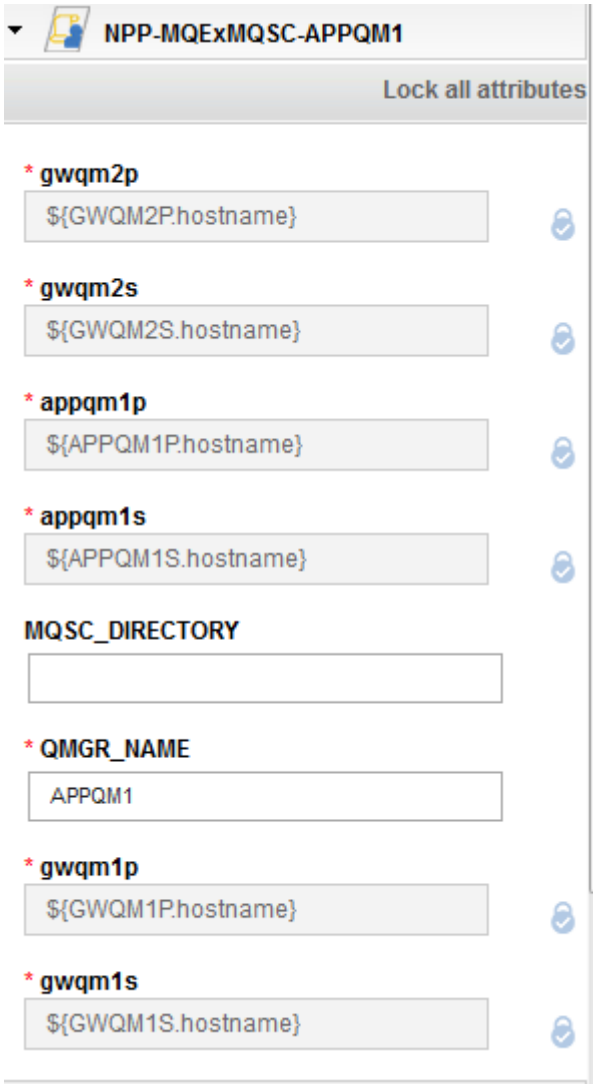
Source code

```
echo "modifiying ipaddress for gwqm1p"
sed -i s/gwqm1p/$gwqm1p/g PAGCLCHL.mqsc

echo "modifiying ipaddress for gwqm1s"
sed -i s/gwqm1s/$gwqm1s/g PAGCLCHL.mqsc

./execute.sh "runmqsc -n < /tmp/mq/mqsc/PAGCLCHL.MQSC"
```

NPP-MQExMQSC-APPQM1-201

The IP addresses of connect nodes in the pattern are capture via the pattern wiring and fed into the script package. The placeholders appqm1s, appqm1p,gwqm1s,gwqm1p are replaced with the actual IP addresses in the MQSC file used to configure the queue manager by running the runmqsc command. In this way all MQ channels resolve to the correct IP addresses for all primary and standby MQ nodes in the pattern.

Parameters



MQSC File

```
ALTER QMGR +
  CCSID(850) +
  CLWLUSEQ(LOCAL) +
  DEADQ('SYSTEM.DEAD.LETTER.QUEUE') +
  CHLAUTH(DISABLED) +
  FORCE

DEFINE QLOCAL('RQ1') +
 CLUSTER('NPPCLUSTER') +
 DEFPSIST(YES) +
 DEFBIND(NOTFIXED) +
```

```
  DISTL(NO) +
  MAXDEPTH(5000) +
  REPLACE
DEFINE QLOCAL('SRA') +
  CLUSTER('NPPCLUSTER') +
  DEFBIND(NOTFIXED) +
  DISTL(NO) +
  MAXDEPTH(5000) +
  REPLACE
DEFINE QLOCAL('SRB') +
  CLUSTER('NPPCLUSTER') +
  DEFBIND(NOTFIXED) +
  DISTL(NO) +
  MAXDEPTH(5000) +
  REPLACE
DEFINE CHANNEL('TO.APPQM1') +
  CHLTYPE(CLUSRCVR) +
  CLUSTER('NPPCLUSTER') +
  CONNAME('appqm1p(1414),appqm1s(1414)') +
  DISCINT(6000) +
  MCATYPE(THREAD) +
  TRPTYPE(TCP) +
  REPLACE
DEFINE CHANNEL('TO.GWQM1') +
   CHLTYPE(CLUSSDR) +
   CLUSTER('NPPCLUSTER') +
   CONNAME('gwqm1p(1414),gwqm1s(1414)') +
   DISCINT(6000) +
   MCATYPE(THREAD) +
   TRPTYPE(TCP) +
   REPLACE
```

```
DEFINE CHANNEL('TO.GWQM2') +
  CHLTYPE(CLUSSDR) +
  CLUSTER('NPPCLUSTER') +
  CONNAME('gwqm2p(1414),gwqm2s(1414)') +
  DISCINT(6000) +
  MCATYPE(THREAD) +
  TRPTYPE(TCP) +
  REPLACE
DEFINE LISTENER('LISTENER.TCP') +
  TRPTYPE(TCP) +
  CONTROL(QMGR) +
  PORT(1414) +
  REPLACE
DEFINE CHANNEL(APPQM1) +
        CHLTYPE(SVRCONN) +
        TRPTYPE(TCP) +
        REPLACE

DEFINE CHANNEL(APPQM1) +
        CHLTYPE(CLNTCONN) +
        TRPTYPE(TCP) +
        CONNAME('appqm1p(1414),appqm1s(1414)') +
        QMNAME(APPQM1) +
        REPLACE
ALTER CHL(APPQM1) CHLTYPE(SVRCONN) MCAUSER('mqm')
ALTER CHL(SYSTEM.DEF.SVRCONN) CHLTYPE(SVRCONN) MCAUSER('mqm')
ALTER QMGR CHLAUTH(DISABLED)
ALTER AUTHINFO(SYSTEM.DEFAULT.AUTHINFO.IDPWOS) AUTHTYPE(IDPWOS) CHCKCLNT(NONE)
REFRESH SECURITY
```

Source code

```
echo "modifiying ipaddress for gwqm1p"
sed -i s/gwqm1p/$gwqm1p/g APPQM1noSYS.mqsc
```

```
echo "modifiying ipaddress for gwqm1s"
sed -i s/gwqm1s/$gwqm1s/g APPQM1noSYS.mqsc

echo "modifiying ipaddress for gwqm2p"
sed -i s/gwqm2p/$gwqm2p/g APPQM1noSYS.mqsc

echo "modifiying ipaddress for gwqm2s"
sed -i s/gwqm2s/$gwqm2s/g APPQM1noSYS.mqsc

echo "modifiying ipaddress for appqm1p"
sed -i s/appqm1p/$appqm1p/g APPQM1noSYS.mqsc

echo "modifiying ipaddress for appqm1s"
sed -i s/appqm1s/$appqm1s/g APPQM1noSYS.mqsc


  ./execute.sh "runmqsc $QMGR_NAME < \"$f\""
```

NPP-MQExMQSC-APPQM2-201

See NPP-MQExMQSC-APPQM1-201 above. APPQM2 has its own unique .mqsc file

NPP-MQExMQSC-GWQM1-201

See NPP-MQExMQSC-APPQM1-201 above. GWQM1 has its own unique .mqsc file

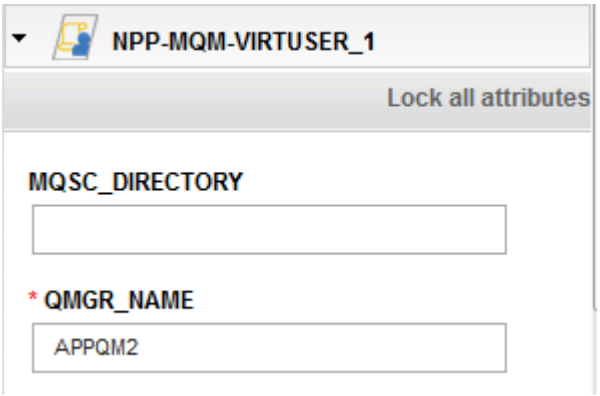
NPP-MQExMQSC-GWQM2-201

See NPP-MQExMQSC-APPQM1-201 above. GWQM2 has its own unique .mqsc file

NPP-MQM-VIRTUSER

This script package address a problem in IIB v10.0.0.3 Standby node that is resolved in IIB v10.0.0.4. If you are at 10.0.0.4 you do not need to use it.

Parameters



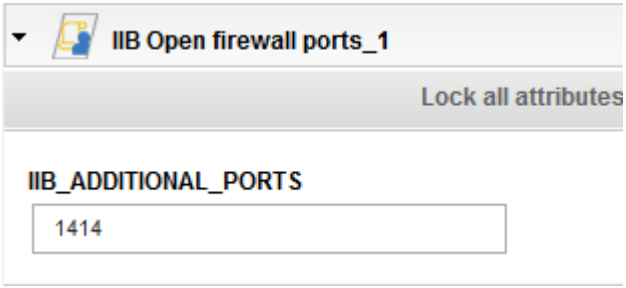
Source code

```
  echo "@DA add virtuser to the mqm group"
usermod -a -G mqm virtuser
```

OpenfirewallPorts-10.0.0.0

Open any additional fire wall ports. May not be required as NPP-Firewall was used to open 1414 for these IIB nodes. Use it if you have other IIB ports you need to open.

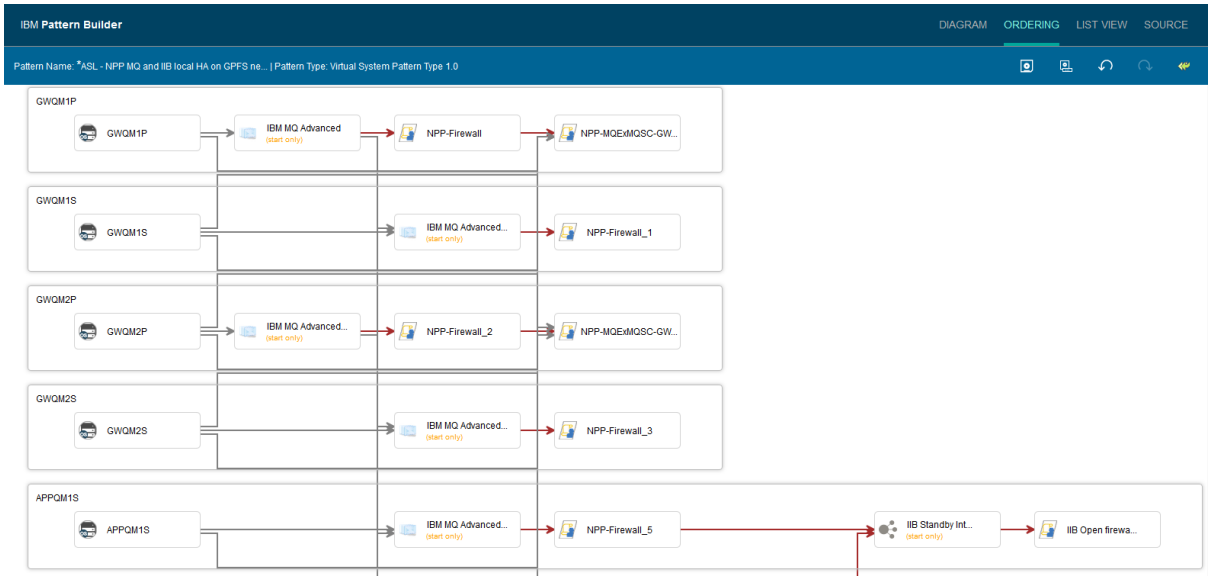
Parameters

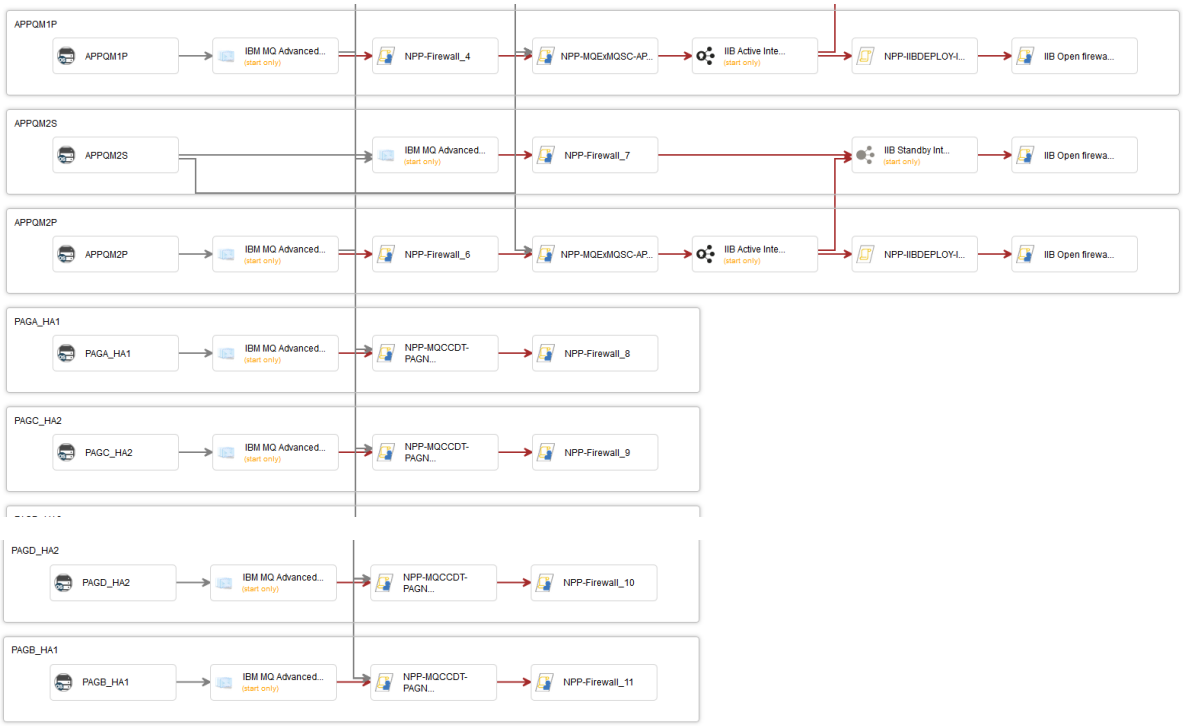


Ordering

Ordering is important such that IP addresses etc can be resolved when script packages to configure MQ channels are run.

Refer to the ordering tab in the IBM Pattern builder for the pattern.





IBM GPFS Shared Service

The pattern has been tested with in single rack against GPFS shared service and we have also tested it against a GPFS service that spanned 2 racks where no Queue Manager primary node and standby were located in the same rack.

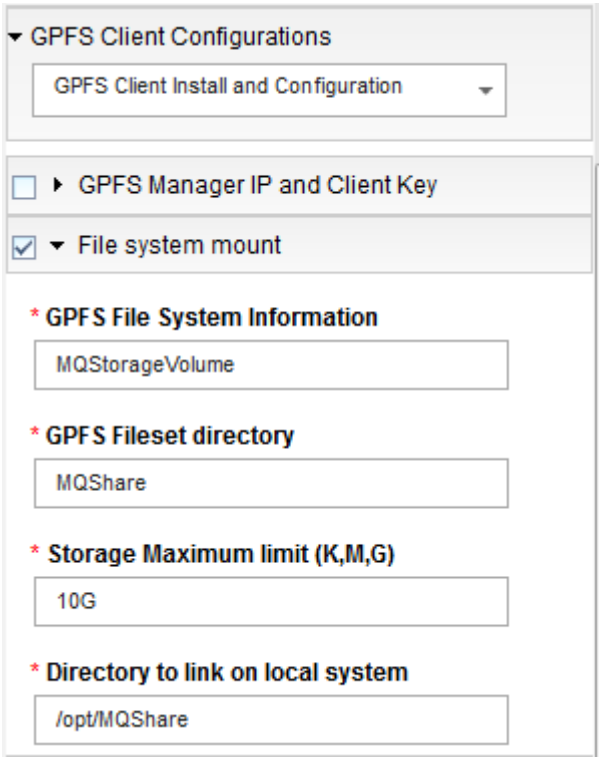
For the twin rack approach we used multi-rack deploy and location profiles to determine that standbys ran in opposite racks from their primaries. In this way we had MQ across rack multi-instance HA. We did this testing before the IIB Primary and Standby parts became available, so IIB multi-instance has not been tested in the same way to date by the team producing this collateral.

GPFS Pattern Type 1.2

IBM Shared Service for GPFS

Application ID:	a-d3481a6b-9ce0-4384-8842-b25fbbf74e6a
Description:	Provides access to a GPFS Cluster Manager where the workload requests a highly available file system.
Service version:	1.2.4.0
Created by:	antonyp
Created on:	Jan 14, 2016, 6:17:00 PM
Supported clients version:	[1.0,1.2]
Pattern type:	GPFS Pattern Type 1.2
Service Type:	External
In the cloud now:	IBM Shared Service for GPFS (Build-CG)

GPFS client configuration



IBM Integration Bus Message Flows

There is a Project Interchange supplied in the materials called NPPMQLocal.zip