IBM Software Group

HADR

High Availability Disaster Recovery

DB2 Information Management Software

Kari Hirvonen
hika@fi.ibm.com
19.08.2004

Disclaimer & Trademarks

The information in this presentation may concern new products that IBM may or may not announce. Any discussion of OEM products is based upon information which has been publicly available and is subject to change. The specification of some of the features described in this presentation may change before the General Availability date of these products.

REFERENCES IN THIS PUBLICATION TO IBM PRODUCTS, PROGRAMS, OR SERVICES DO NOT IMPLY THAT IBM INTENDS TO MAKE THESE AVAILABLE IN ALL COUNTRIES IN WHICH IBM OPERATES.

IBM MAY HAVE PATENTS OR PENDING PATENT APPLICATIONS COVERING SUBJECT MATTER IN THIS DOCUMENT. THE FURNISHING OF THIS DOCUMENT DOES NOT IMPLY GIVING LICENSE TO THESE PATENTS.

TRADEMARKS

The following terms are registered trademarks of International Business Machines Corporation in the United States and/ or other countries: AIX, AIXwindows, AS/ 400, DB2, e(logo), IBM, IBM(logo), Information Warehouse, Netfinity, NUMA- Q, OS/ 2, OS/ 390, OS/ 400, Parallel Sysplex, PowerPC, PowerPC(logo), RISC System/ 6000, RS/ 6000, S/ 390, Sequent, SP2, System/ 390, The Engines of e- business, ThinkPad, Tivoli(logo), TURBOWAYS, VisualAge, WebSphere.

The following terms are trademarks of International Business Machines Corporation in the United States and/ or other countries: AIX/ L, AIX/ L(logo), AS/ 400e, DB2 OLAP Server, DB2 Universal Database, e- business (logo), HACMP/ 6000, Intelligent Miner, iSeries, Network Station, NUMACenter, PowerPC Architecture, PowerPC 604, POWER2 Architecture, pSeries, Shark, SP, Tivoli Enterprise, TME 10, Videocharger, Visualization Data Explorer, xSeries, zSeries.

A full list of U. S. trademarks owned by IBM may be found at http:// iplswww.nas.ibm.com/wpts/trademarks/trademar.htm.

NetView, Tivoli and TME are registered trademarks and TME Enterprise is a trademark of Tivoli Systems, Inc. in the United States and/ or other countries.

Microsoft, Windows, Windows NT and the Windows logo are registered trademarks of Microsoft Corporation in the United States and/ or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

Oracle is a registered trademark of Oracle Corporation in the United Status and/or other countries.

LINUX is a registered trademark of Linus Torvalds.

Intel and Pentium are registered trademarks and MMX, Itanium, Pentium II Xeon and Pentium III Xeon are trademarks of Intel Corporation in the United States and/ or other countries.

Java and all Java- based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/ or other countries.

Other company, product and service names may be trademarks or service marks of others.

Agenda

- **The Basics**
  - Architecture
  - Setup
- **The Details**
  - Failover / Failback
  - Administration
  - Performance
- **The Specials**
  - Things To Remember
  - Limitations
- **The Demo**
  - Functional Live Demo ...
- **The Complements & Alternatives**
  - Client Reroute
  - Q-Replication

DB2 Information Management Software

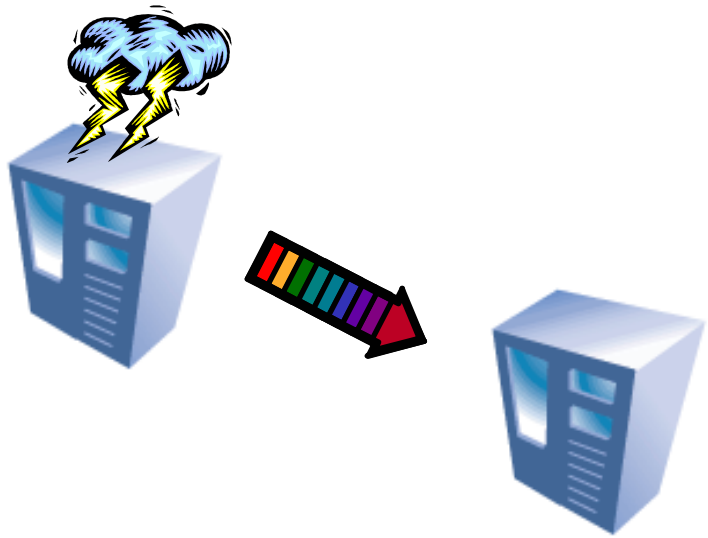
Presentation based on slides kindly provided by

Angel González, IBM Germany

Main Goals of the Design

- **Ultra-fast failover**
- **Easy administration**
- **Handling of site failures**
- **Negligible impact on performance**
- **Configurable degree of consistency**
- **Protection against errant transactions**
- **Software upgrades without interruption**
- **Very easy integration with HA-software**
- **Eventually, no need for HA-software at all**
- **Transparent failover and failback for applications (combined with “client re-route”)**

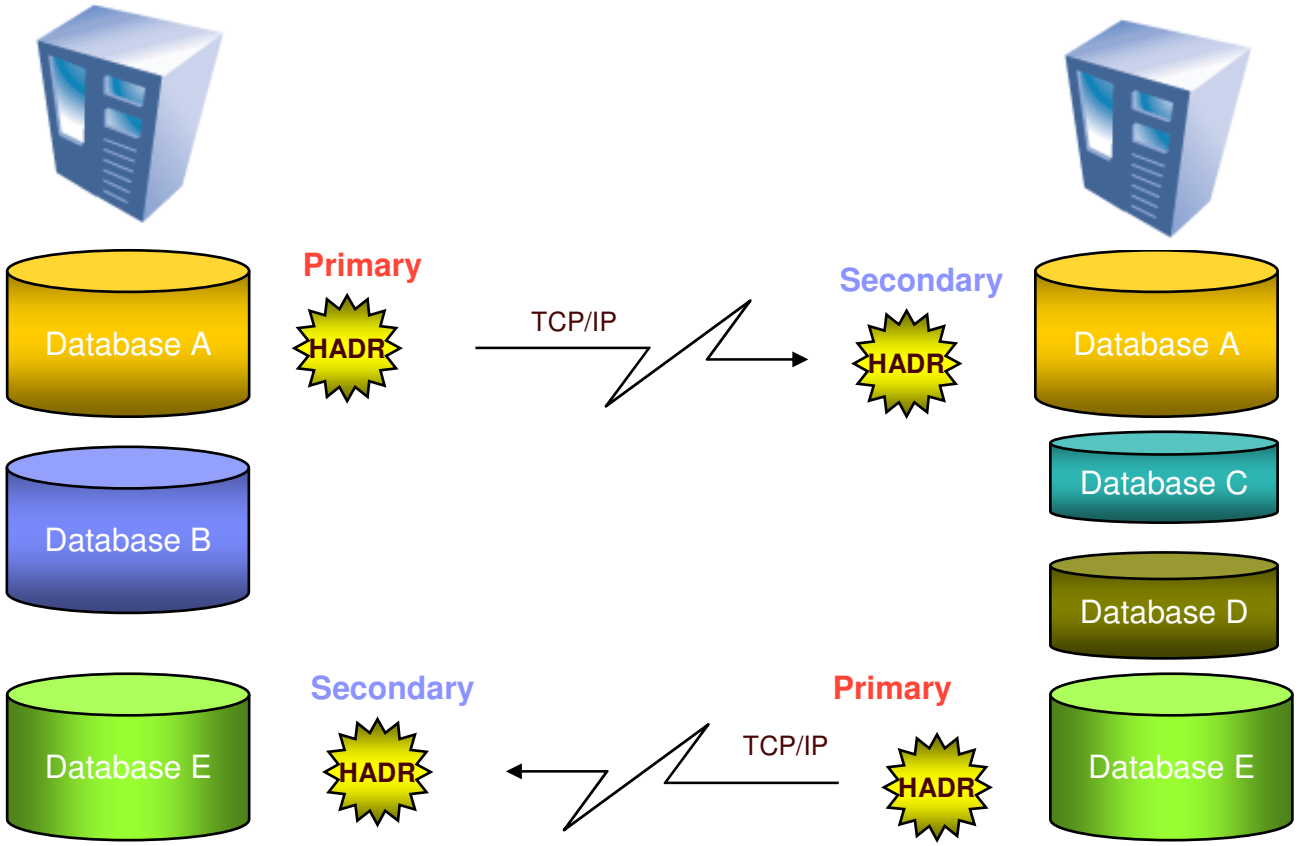
Basic Principles of HADR



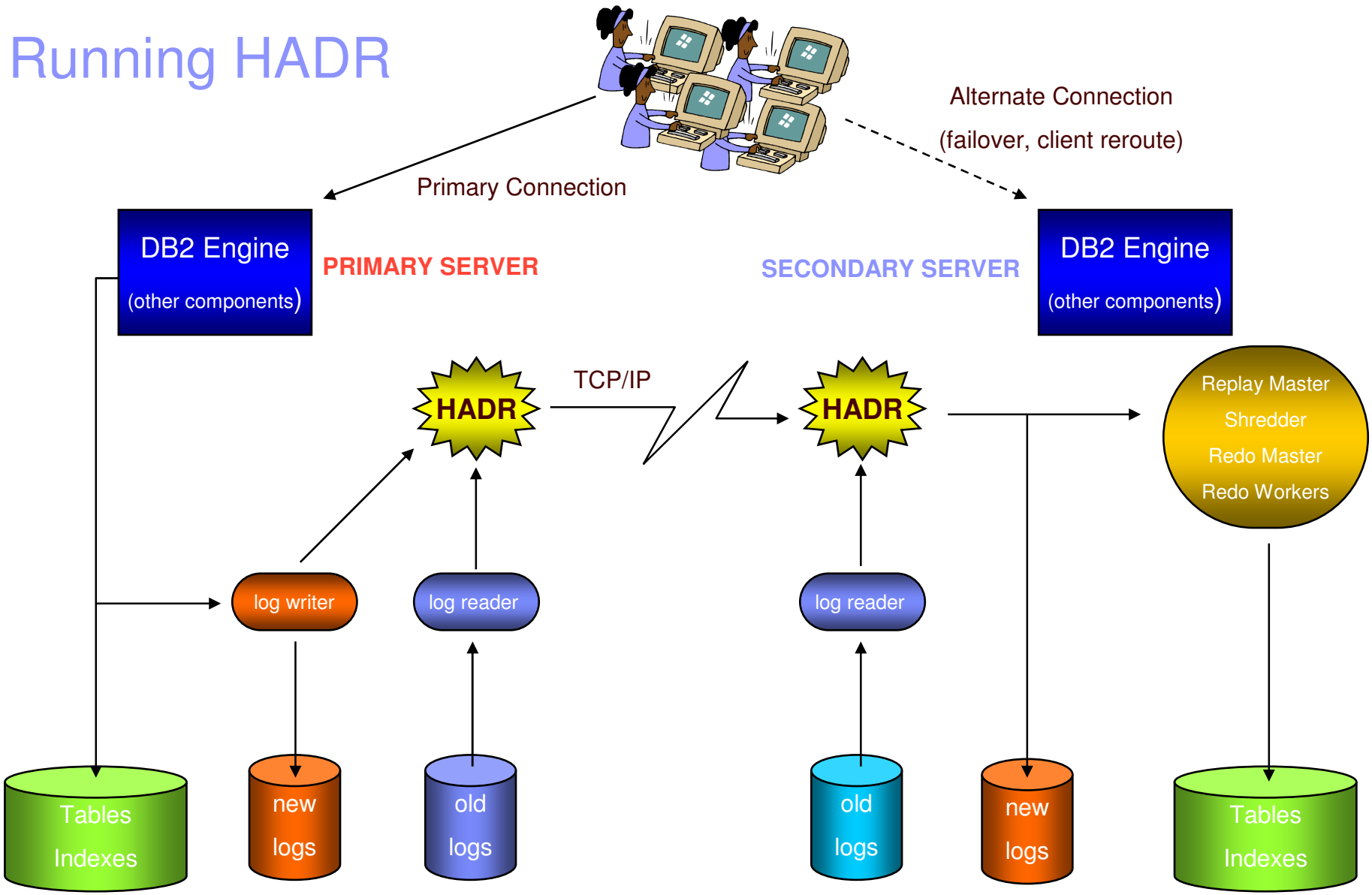
- **Two active machines**
  - Primary
    - Processes transactions
    - Ships log entries to the other machine
  - Secondary
    - Cloned from the primary
    - Receives and stores log entries from the primary
    - Re-applies the transactions
- **If the primary fails, the secondary can take over the transactional workload**
  - The secondary becomes the new primary
- **If the failed machine becomes available again, it can be resynchronized**
  - The old primary becomes the new secondary

Scope of Action

- **HADR replication takes place at the database level.**



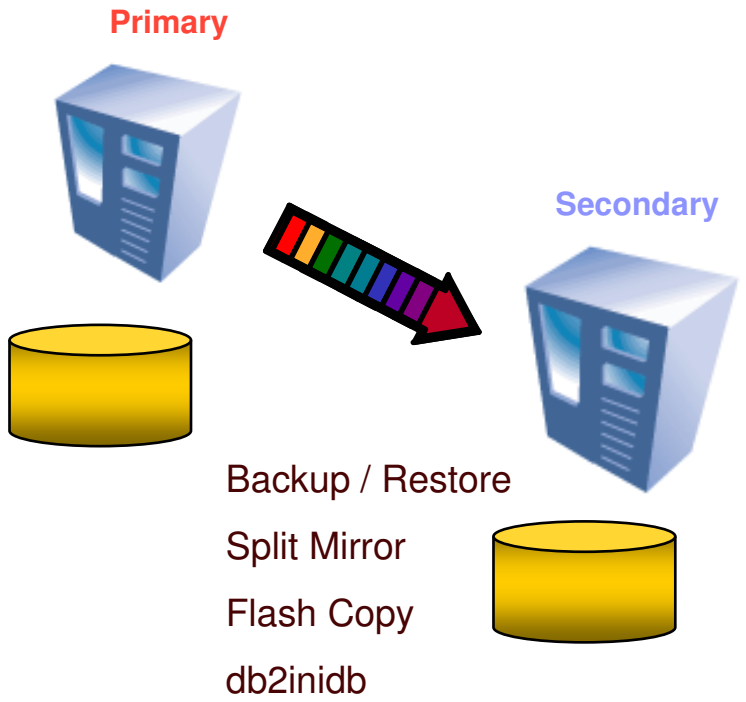
Running HADR



HADR Setup: Three steps (1)

1. Clone the primary

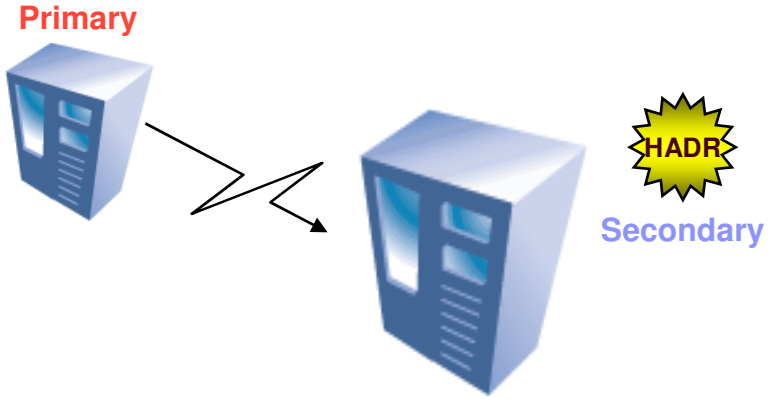
- The target database (referred to as a secondary) is created using DB2's restore facility, flash copy, or by split mirror
- After the restore, the secondary is placed in "perpetual rollforward" mode
- Strict symmetry of table space and container configuration is required on the secondary. Name, path, size all must match. Relative container paths are allowed, and the full path may differ in this case.
  - If not, HADR may fail to replicate a container operation on the secondary. In such a case replication of the affected tablespace stops and the tablespace will be left in "rollforward in progress" .



2. Start the secondary

3. Start the primary

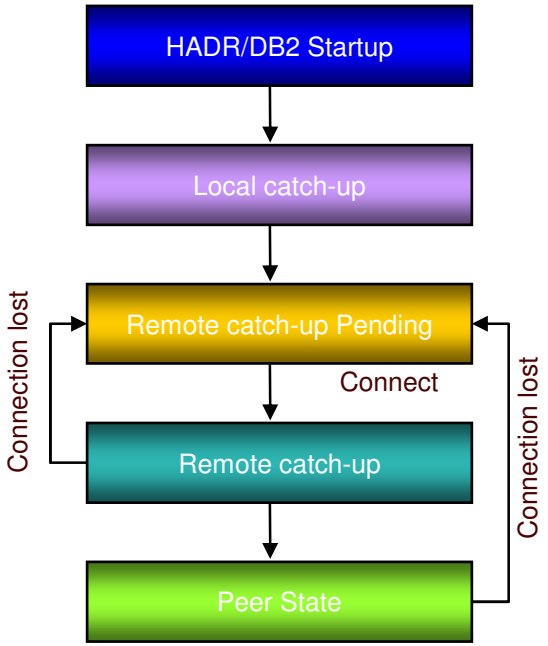
HADR Setup: Three steps (2)



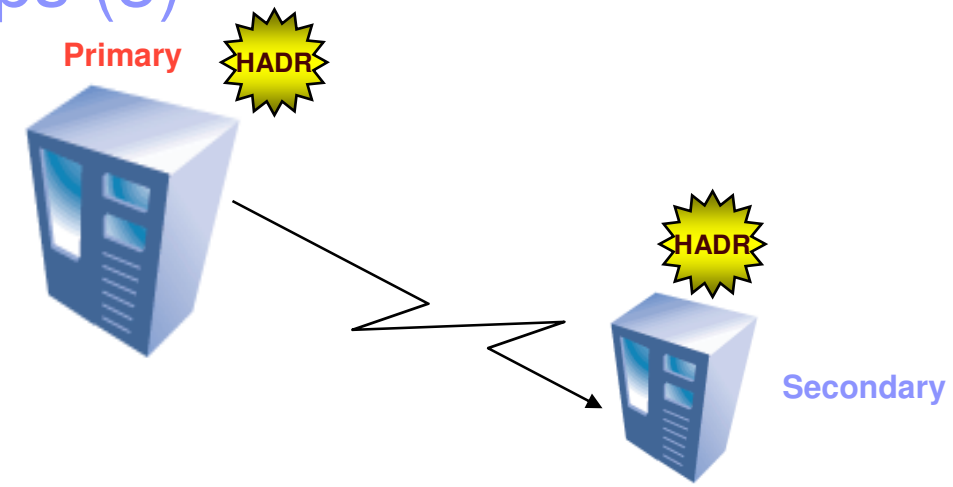
1. **Clone the primary**

2. **Start the secondary**
   - Local catch-up
     - Replay locally available log files (if any)
     - Replay log files that may be retrieved at the primary and shipped to the secondary
   - Remote catch-up (primary must be available)
     - Replay log pages from the primary's **archived** logs
     - Replay log pages from the primary's **active** log to the secondary until the secondary is caught up to the tail of the log.
     - Replay log pages from the primary's **in-memory** log buffer to the secondary whenever the primary flushes those pages to disk
   - Peer State (primary must be available)
     - Primary and secondary work in sync

3. **Start the primary**



HADR Setup: Three steps (3)

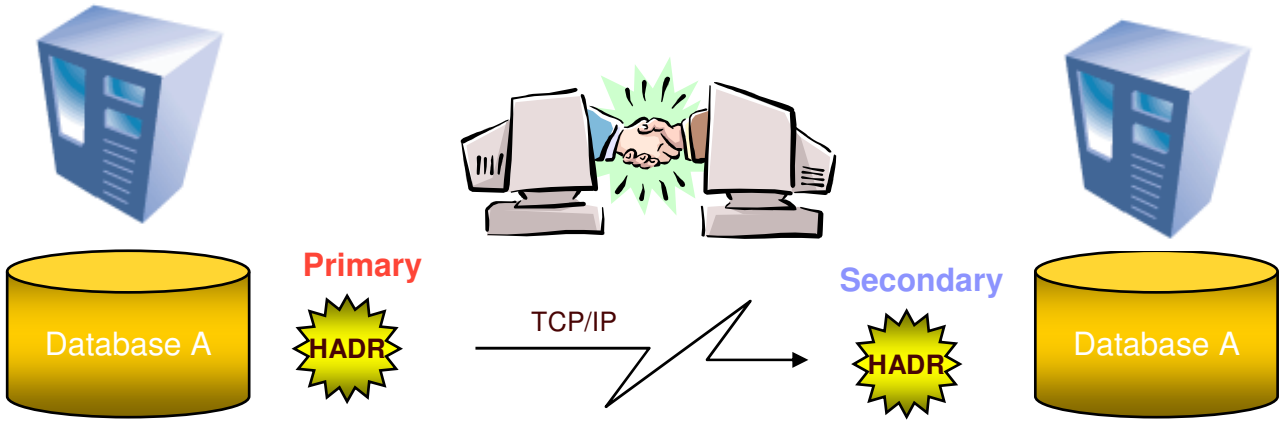


1. **Clone the primary**
2. **Start the secondary**
3. **Start the primary**

- Unless started "`by force`", the HADR primary waits for the secondary to contact it.
- If the secondary does not show up after a waiting period (configurable), HADR will not start.
- This behavior avoids that two systems could work as primary at the same time (split brain).

Peer State

- HADR is ready to fail over to the secondary
- HADR retrieves log data from the primary's in-memory log buffer cache at the time log pages from the buffer are flushed to the primary's log disk
  - Not only "commit" flushes the log buffer

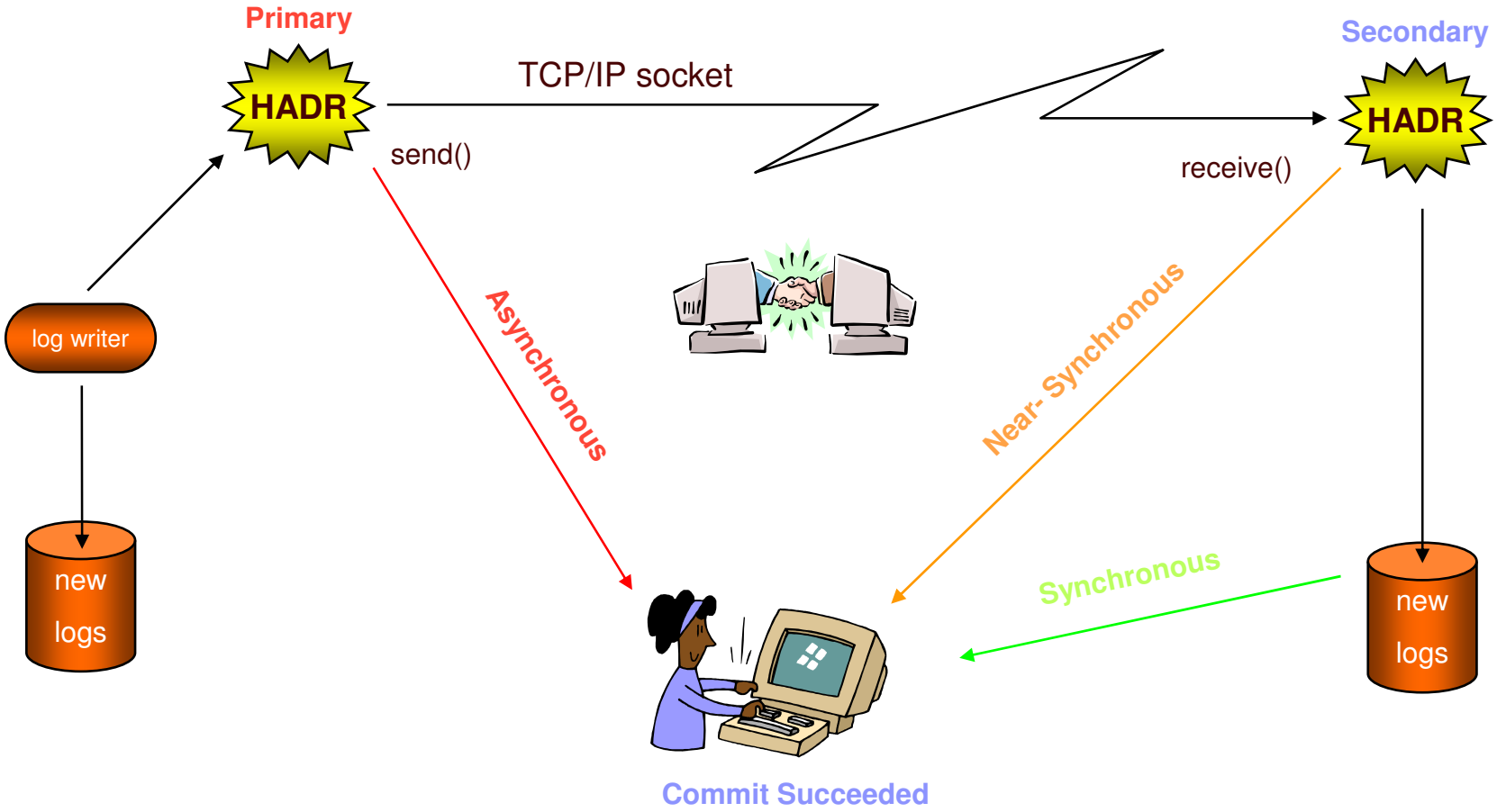


Synchronization modes

- **Synchronous (zero data loss)**
  - The log data is flushed to stable storage at the secondary before committing on the primary.
  - The log flush and TCP send() – receive() will be serialized.
  - The secondary will not write the log until it hears from the primary that that the same log is on disk there.
  - The primary cannot proceed on to the next log flush until it receives the ACK notifying it that the secondary has written the log.
  - A commit succeeds when the log data is **on disk** at the primary and at the secondary.
- **Near synchronous**
  - The log data has successfully been sent to the secondary site but it might not have been flushed to stable storage when the commit on the primary succeeds.
  - The log write at the primary and the send to the secondary are performed in parallel at the primary.
  - A commit succeeds when the log data is **on disk** at the primary and it has been **received** by the secondary
- **Asynchronous**
  - The log data has been given to TCP/IP and the socket send call returned successfully.
  - This is not an acknowledgment of its successful receipt by the secondary, so if the connection breaks, the secondary may not have received everything the primary sent.
  - However, TCP/IP sockets do guarantee delivery order, so while the socket stays alive there will never be missing or out-of-order packets as seen by the secondary.
  - The log write at the primary and the send to the secondary are performed in parallel at the primary.
  - A commit succeeds when the log data is **on disk** at the primary and it has been **sent** to the secondary

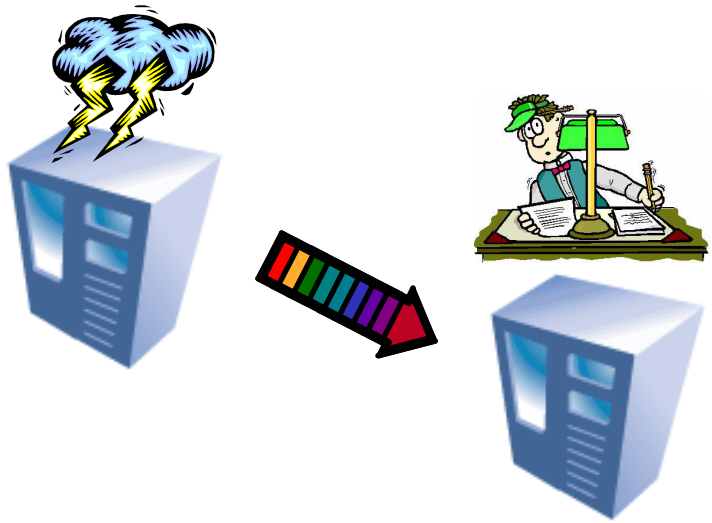
Synchronization modes

- **Synchronous, Near Synchronous, Asynchronous**



Failover

- **Single command called "TAKEOVER"**
  - Change the secondary into a primary in response to a failure of the primary (normal failover).
  - Switch the roles of a healthy primary-secondary pair
  - No more db2start / restart database / rollforward etc.
    - That was the "*old fashioned*" procedure with HACMP, SunCluster, MS-Cluster Services, etc.
- **It can be embedded in very simple heart-beat scripts**
- **In the future, a heart-beat and a coordinator will be shipped**
- **For the client applications, the automatic client re-route will provide transparent failover**



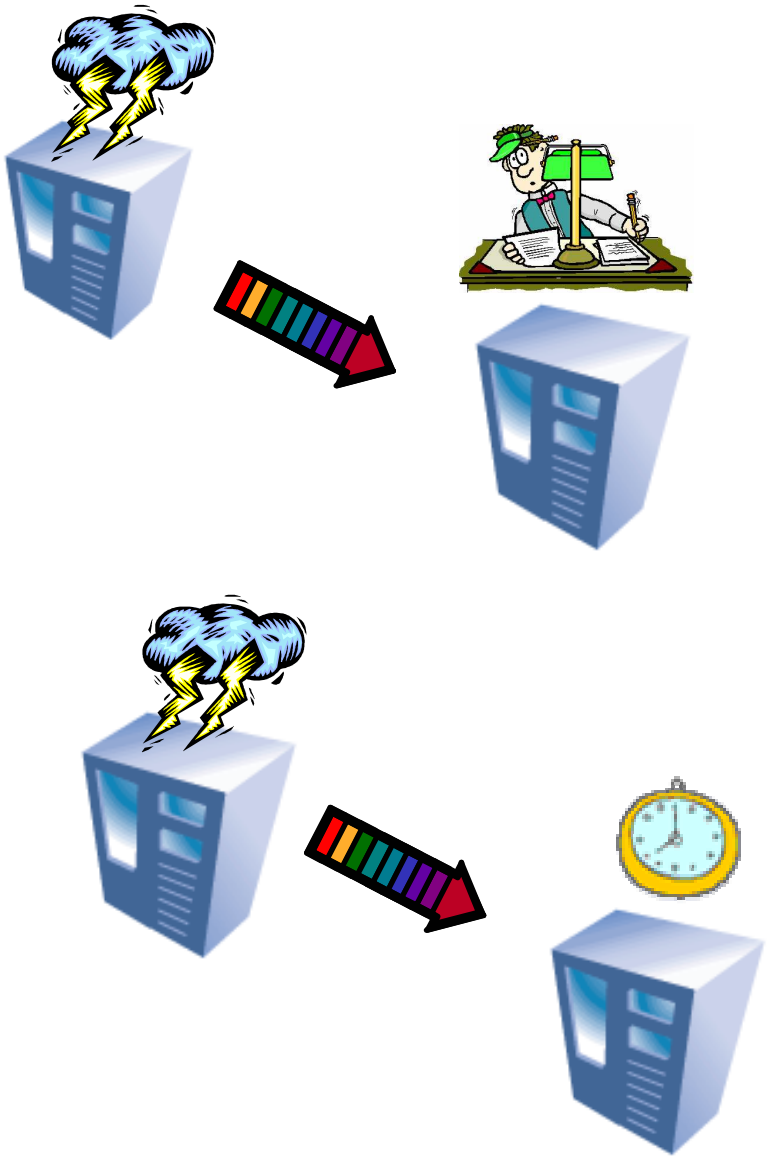
Two flavors of TAKEOVER

- **Emergency TAKEOVER (by force)**
  - The secondary sends a notice asking the primary to shut itself down.
  - The secondary does NOT wait for any acknowledgement from the primary to confirm that it has received the takeover notification or that it has shut down
  - The secondary stops receiving logs from the primary, finishes replaying the logs it has already received, and then becomes a primary.

- **Normal TAKEOVER (no force)**
  - Primary and Secondary switch roles. The following steps are carried out:
    1. Secondary tells primary that it is taking over.
    2. Primary forces off all client connections and refuses new connections.
    3. Primary rolls back any open transactions and ships remaining log, up to the end of log, to secondary.
    4. Secondary replays received log, up to end of the log.
    5. Primary becomes new secondary.
    6. Secondary becomes new primary

Errant Transactions

- **First release of HADR**
  - Setup HADR and bring it to peer state
  - Suspend the secondary for a while
  - If a transaction messes up the database on the primary:
    - Shutdown the primary
    - Failover to the secondary
  - If everything goes well:
    - Let the secondary catch up the primary
    - Suspend the secondary for a while, repeat the loop
- **Planned improvement**
  - Configurable delay for applying changes on the secondary



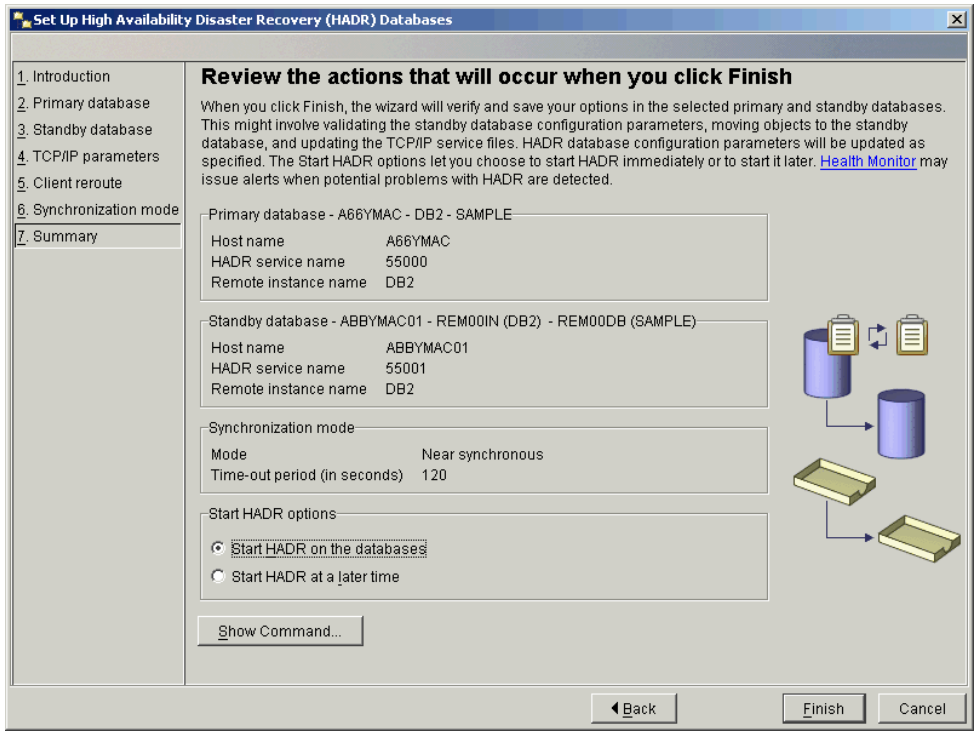
Software upgrades on the fly

1. **HADR in peer state**
2. **Suspend the secondary for a while**
   - Stop HADR
3. **Upgrade the secondary**
4. **Start the secondary again**
   - Let it catch-up with primary
5. **Issue a normal TAKEOVER**
   - The primary and secondary change their roles
6. **Suspend the new secondary**
7. **Upgrade the new secondary**
8. **Reactivate the new secondary**
   - Let it catch-up with primary
9. **Optionally, TAKEOVER again**
   - The primary and secondary play their original roles

HADR Wizard

- **The Configure HADR Databases Wizard facilitates the process to set up the primary and Secondary databases for HADR**

- **The wizard will guide the user to perform the following tasks:**
  - Identify the HADR pair
  - Prepare the primary database for log shipping
  - Perform database backup
  - Copy backup image to secondary server
  - Perform restore on the selected Secondary database
  - Move any database objects not included in the backup image
  - Update service files
  - Update HADR related configuration parameters on both databases
  - Provide an option to Start HADR



Configuration

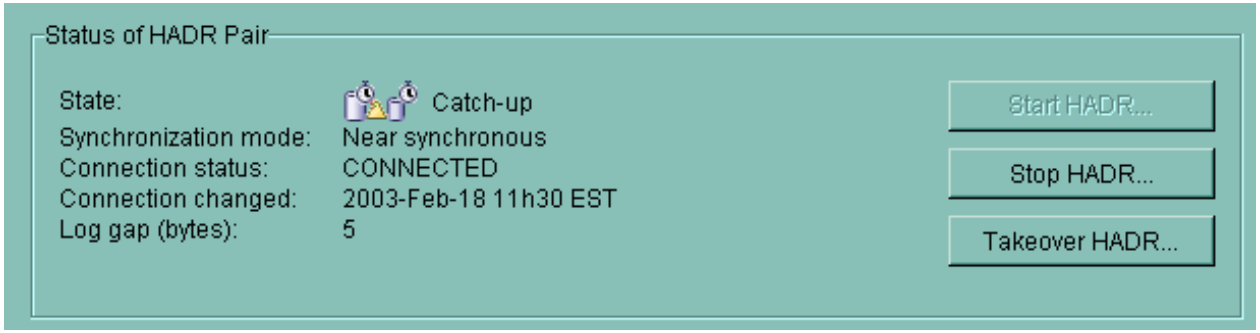
- **Setting these parameters neither enables HADR on a database nor indicates whether a database is primary or secondary.**
- **This separation allows users to disable HADR without losing HADR configuration**

```
db2set DB2_HADR_BUF_SIZE = <buffer size>

db2 "update db cfg for database using…"
```

```
HADR_LOCAL_HOST         # Either a host name or an IP address
HADR_LOCAL_SVC          # Either a service name or a port number

HADR_REMOTE_HOST        # Either a host name or an IP address
HADR_REMOTE_SVC         # Either a service name or a port number

HADR_REMOTE_INST        # Instance name

HADR_TIMEOUT            # In seconds

HADR_SYNCMODE           # One of:
                        # SYNC       (for synchronous)
                        # ASYNC      (for asynchronous)
                        # NEARSYNC (for near-synchronous)
```

HADR Action Windows

- **A new set of tools are available to view the status of an HADR pair. From this view the following actions can be performed:**
  - Refresh the status information
  - Stop and Start HADR
  - Takeover HADR
- **The GUI will enhance the CLP by allowing the current role and state of a database to be used to determine which commands and options are valid before submitting the command to the engine**



HADR Monitor

- **When a snapshot is taken on an HADR node, it includes log positions of both primary and secondary. The two machines will exchange their log positions via messaging.**
- **In the case of broken connection, remote log position and log gap will not be updated, but the old values will still be reported, so that users have some idea of the state of the system when the connection was lost.**
- **If there has not been a connection since the HADR EDU was started, no remote log position or log gap will be reported.**

```
GET SNAPSHOT FOR DATABASE ON database_alias
HADR status
  Role                   = Primary
  State                  = Peer
  Synchronization mode   = Sync
  Connection status      = Connected, 11-03-2002 12:23:09.35092
  Heartbeat missed       = 0
  Local host             = host1.ibm.com
  Local service          = hadr_service
  Remote host            = host2.ibm.com
  Remote service         = hadr_service
  Remote instance        = dbinst2
  timeout(seconds)       = 120
Primary log position(file, page, LSN) = S0001234.LOG, 12, 0000000000BB800C
Standby log position(file, page, LSN) = S0001234.LOG, 12, 0000000000BB800C
Log gap running average(bytes) = 8723
```

Performance Considerations

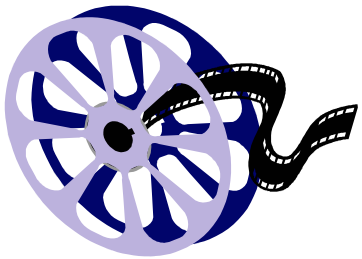
- **Theoretically, the secondary can process transactions faster than the primary because it does not need to perform locking, manage connections, or generate logs.**
- **On the HADR primary, the commit processing may be slightly delayed if the system is configured to use a synchronization mode of SYNC or NEARSYNC.**
- **If the connection is lost, a transaction may experience a commit delay of about the length of the configured HADR_TIMEOUT**
- **The performance of DB2 functionality immediately after failover may not be exactly the same as it was just prior to the failover.**
  - A short ramp-up time should be expected.
- **The best way to describe the network requirements is to monitor logging activity and determine the rate of logging. Whatever the rate of logging is, that would be the network transfer rate required**
  - In a future release, HADR may ship substantial amounts of data between the HADR partner systems in order to replicate certain non-logged operations
  - Alter table log index will increase the logging requirements
- **If the user chooses to not log index builds, then those indexes will need to be rebuilt on the new HADR primary after a takeover.**

Backup / Restore Considerations

- **Backup is possible on the primary**
- **Backup operations are not supported on the Secondary database**
  - HADR does not replicate the recovery history file
- **Database-level restores are allowed against existing HADR primary and HADR secondary databases.**
  - As part of the restore operation, however, the database will be changed from its existing HADR role into a “standard” DB2 database. As a result, after a restore the database will not automatically reconnect with its HADR partner.
- **Tablespace-level restores on the primary and secondary will be disabled**
- **Redirected restores are supported at the database level**

Other considerations

- **HADR database must not use circular logging**
  - Log archiving methods supported by DB2 are: disk, TSM, vendor, userexit and logretain However, log archiving is done only from the primary database.
  - The HADR secondary does not archive log files, it deletes them when they are not longer needed (the primary says when)
- **Log mirroring (`MIRRORLOGPATH`) works as usual on both the primary and the Secondary**



- **HADR does NOT replicate configuration parameters**
  - In future releases, HADR config may be managed by an HADR coordinator, which may propagate HADR config globally and check consistency
- **HADR does NOT replicate the shared libraries or DLLs of the UDFs or the Stored Procedures**
  - The DB2 administrator must copy the executables in the proper directories

Limitations of the first release

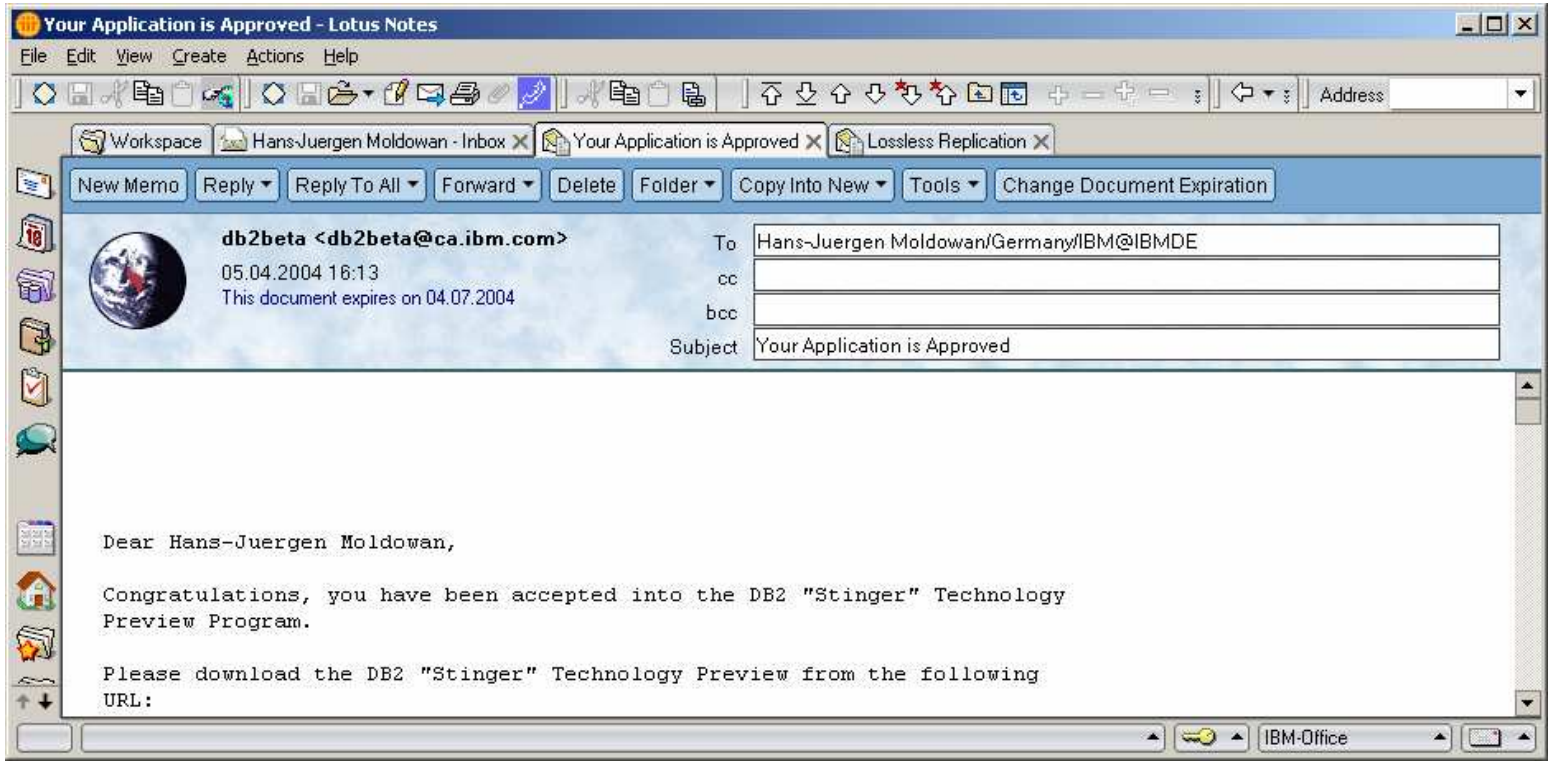
- **Unsupported operations**
  - Non-Logged Operations
  - Use of Infinite Logging
  - Data Links
  - Log File Backups on the Secondary
  - Tablespace Restores
- **HADR is supported in DB2 ESE (included), WorkGroup and Express Edition (additional fee). It is not supported on DB2 ESE with DPF. (This MAY change soon after GA).**
- **The primary and secondary databases must have the same operating system version and the same version of DB2, except for a short period of time during a rolling upgrade.**
- **The DB2 release on the primary and Secondary databases must be the same bit size (32 or 64 bit).**
- **Backup operations are not supported on the secondary database**
- **No multiple targets - having more than one secondary for a single primary**
- **Automatic failover - utilizing a coordinator to integrate with existing monitoring facilities to trigger failovers**
  - Note that HADR can also be deployed in combination with a cluster manager. The cluster manager would be responsible for fault detection and for driving the failover activities.
- **Reads on the secondary allowing read-only queries to run on the secondary are not allowed.**
- **Load operations with the COPY NO option specified are not supported**

Client Reroute

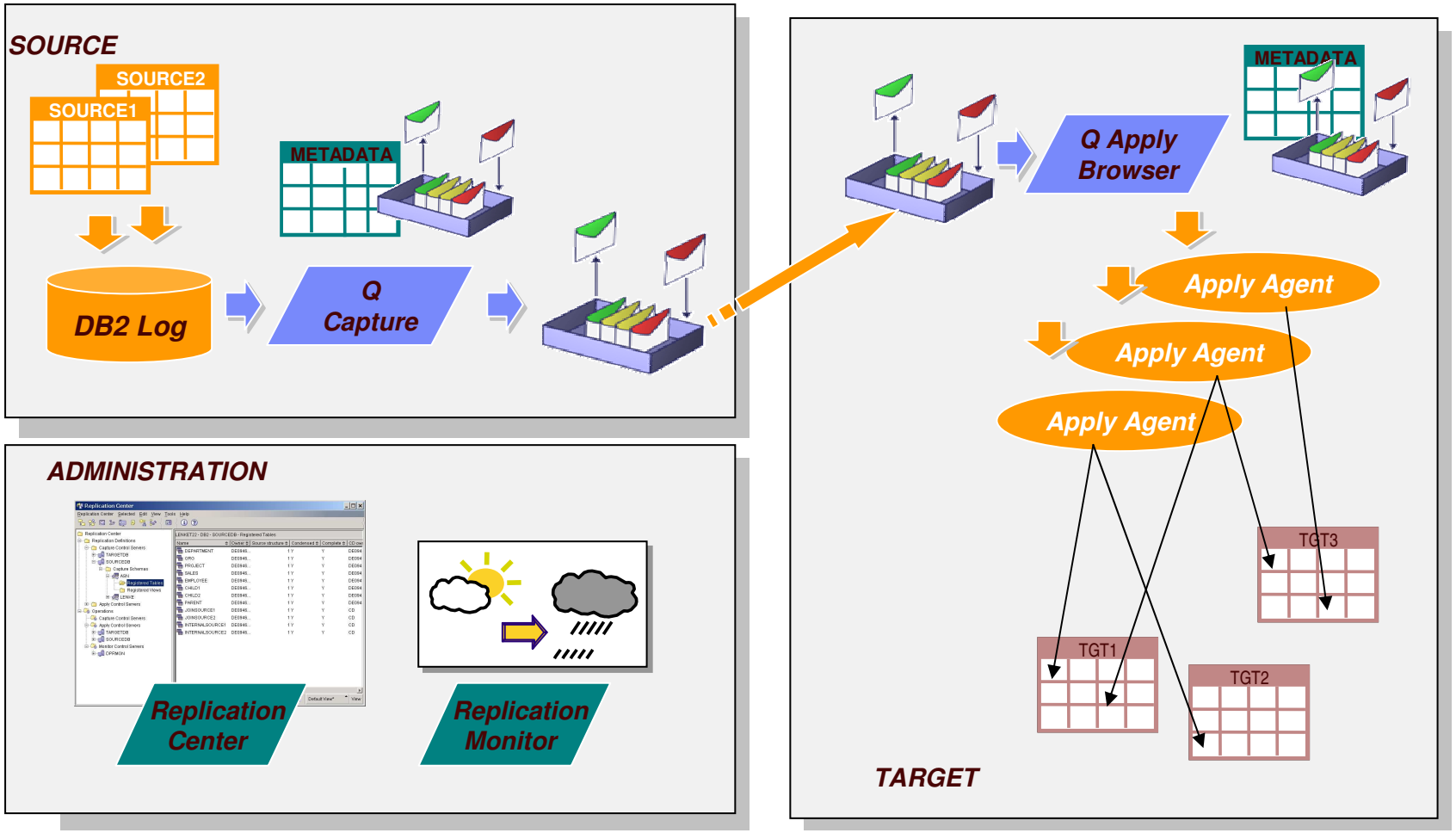
- **Whenever a server crashes, each client that is connected to that server gets a communication error which terminates the connection leading to an application error.**
  - The DB2 UDB client code attempts to re-establish the connection to either the original server or to a new server.
  - When the connection is re-established, the application will receive an error that informs it of the transaction failure, but the application can continue with the next transaction.
- **The Automatic Client Reroute feature could be used in following configuration environments:**
  - Enterprise Server Edition (ESE) / Data Partitioning Feature (DPF)
  - Dpropr (Data Propagator) -style Replication
  - High Availability Cluster Multiprocessor (HACMP)
  - High Availability Disaster Recovery (HADR) environment
- **Also available with LDAP Clients**

Stinger Technology preview

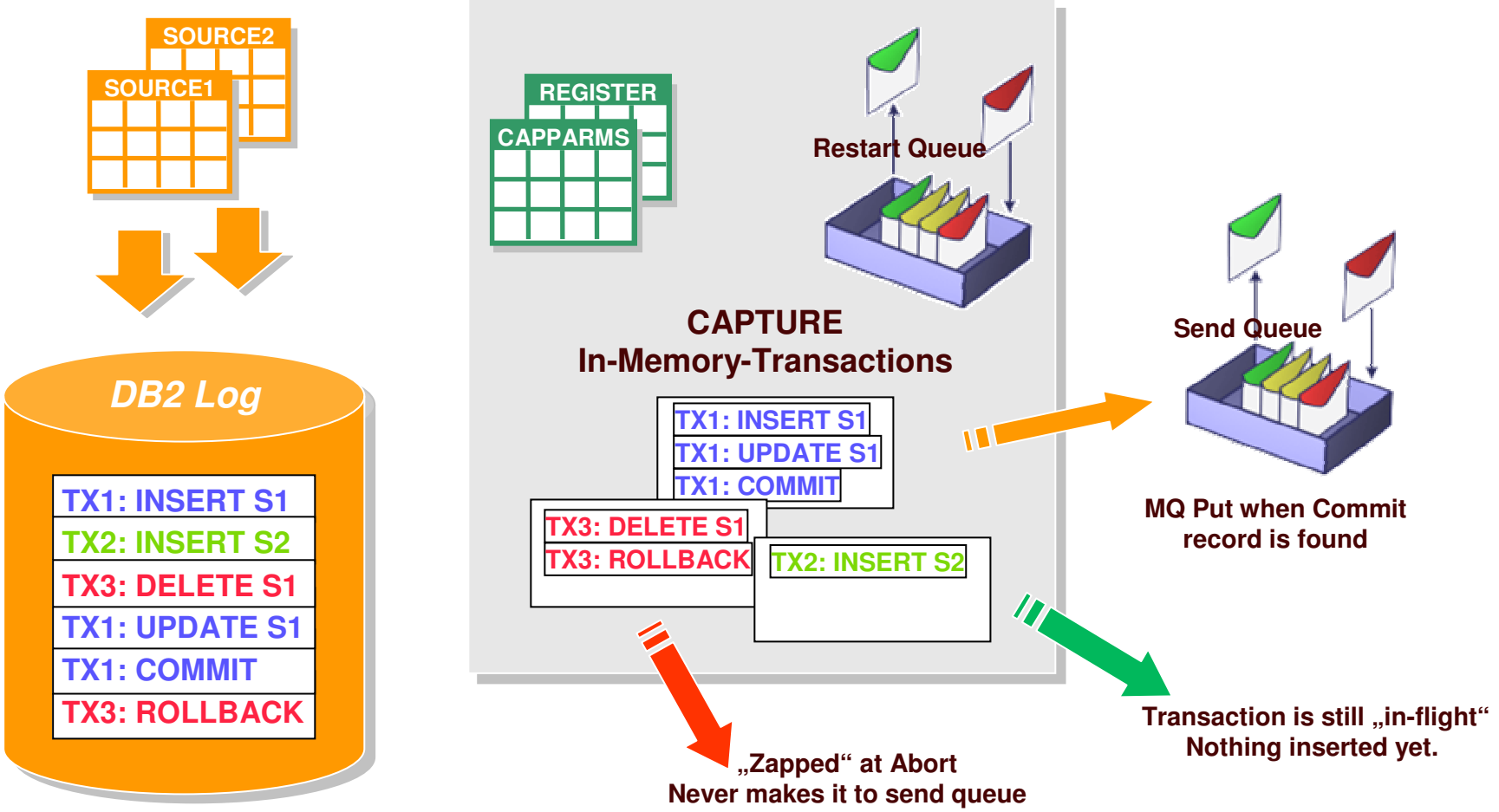
- http://www.ibm.com/software/data/db2/stinger/
- **Fill out the registration**
- **You will get an approval with a download location and key**



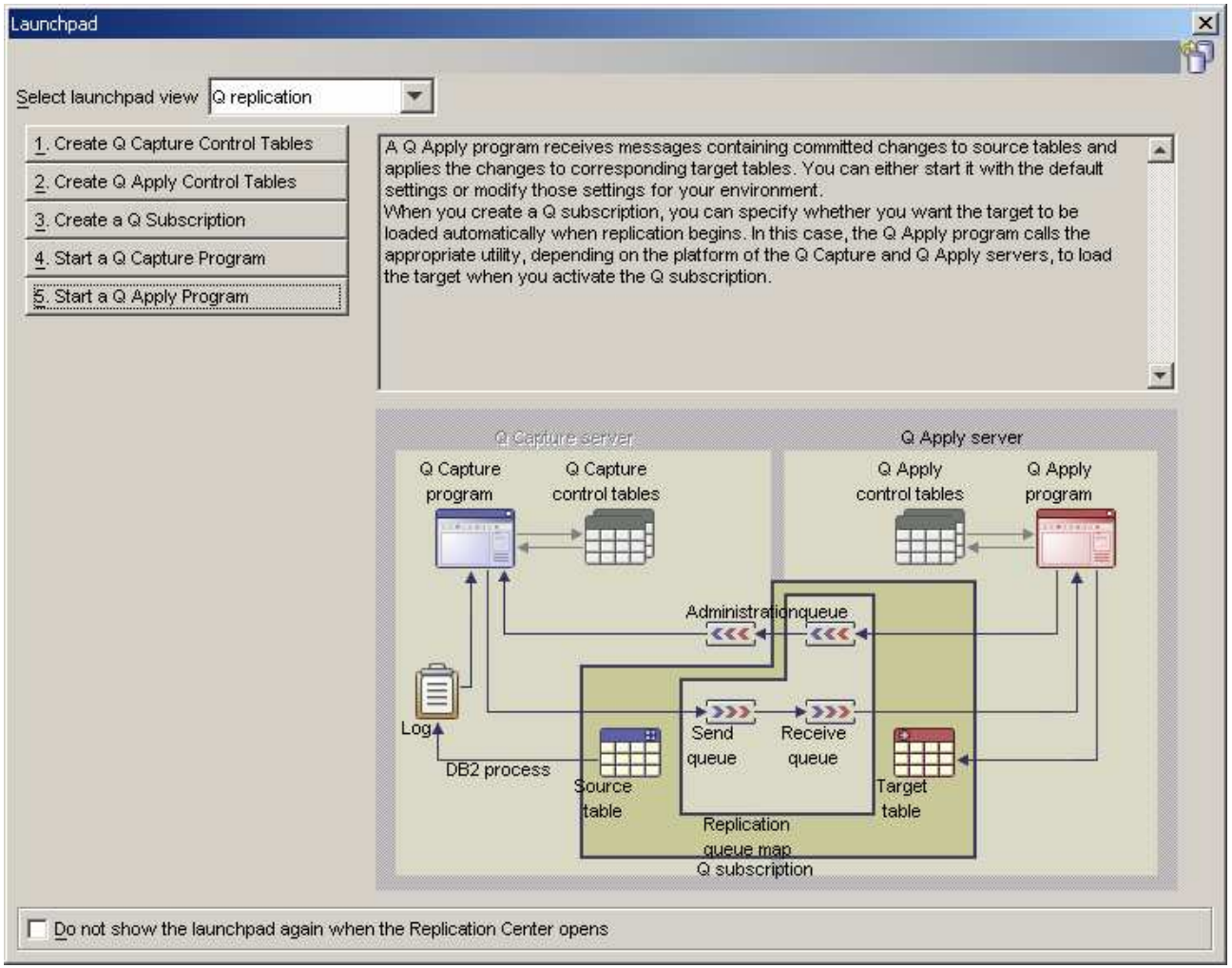
Q-Replication: Architecture



Transactional Capture in Q-Replication



Administration with Q-Replication



Comparing HADR / Q-Replication

Functionality	HADR	Q-Replication
Synchronous "zero-loss" replication*	Yes	No
Asynchronous replication	Yes	Yes
Replicate entire database*	Yes	No (1)
Replicate a subset of database tables	No	Yes
Replicate a subset of data	No	Yes
Cross-OS support	No	Yes
Two-way replication	No	Yes
Multiple targets (standby)	No	Yes
Read/Write on targets (standby)	No	Yes
Automatic Client Reroute	Yes	Yes
Monitoring	Yes	Yes
Requires staging tables	No	No
Requires installation and set-up of WebSphere MQ server	No	Yes
Network compression **and encryption**	No	Yes
DPF Support	No	Yes
(1) While Q-Replication can replicate all the data, each table needs to be replicated individually and other database objects (like table spaces) are not kept in sync. Also,DDL is not replicated.		
*Required for "true high availability"		

Questions and Answers

HADR

High Availability Disaster Recovery

Ja sitten ei muuta kuin hauskaa iltaa!