Effective Clustering of Microsoft BizTalk Server on hp ProLiant Servers

Eddie Martinez

Chris Stewart

ISS Microsoft Infrastructure Solutions

HP WORLD 2003
Solutions and Technology Conference & Expo

Agenda

- BizTalk 2002
  - BizTalk Compute Farms
  - Complex BizTalk Clusters
- BizTalk 2004
  - BizTalk Compute Farms
  - Complex BizTalk Clusters

Clusters

- Term has multiple technical meanings.
- Most folks use the term generically, even when they speak of a specific sort of cluster.
- Two very distinct sorts of clusters:
  - High-availability
  - Compute farms

High Availability Clusters

- Applications are deployed as one or more instances spread across two or more servers that are able to access the same shared storage devices.
- An application so deployed is said to be highly available because every instance of the application can ultimately collapse onto just one of the servers in the cluster should all other servers in the cluster fail
- In case of such a failure, all instances of the application can continue to operate as if no interruption to availability had occurred.

Compute Farms

- Many servers are pooled together so that their collective hardware resources can be used to increase the workload capacity of an entire system.
- Sometimes called server pools or server farms.
- An intelligent availability detector is part of the architecture.
  - Can be internal to the compute farm or external to it.
  - When one server or another in the farm ceases to be available, it no longer receives any additional work to complete.

Compute Farms

- Push model for workload distribution: Load balancing
  - Clients make workload requests of the compute farm.
  - Load balancer simply directs each request to an appropriate server within the farm so that the workload is effectively distributed.
- Pull model for workload distribution
  - A centralized resource houses a collection of pending work that must still be completed.
  - The servers in a compute farm point to the shared resource as the location from which to pull additional workload items.
  - If the execution of any specific workload item on a given server must be temporarily paused, then that server serializes both the data associated with the workload item and the metadata, which indicates that workload item’s state, back to the shared resource.

Combination Clusters

- Other clustering technologies, such as Beowulf clustering and Oracle Parallel Server, incorporate elements from both clustering models.
- These are outside the scope of this discussion.

Pull Workload Distribution

- When the execution is able to continue, both the data and the metadata are de-serialized from the shared resource, and upon completion, the finished job is returned to the shared resource.
- In this pull model, it is interesting to note that no infrastructure beyond the compute farm and the shared resource are required to efficiently distribute workload items across the entire farm.
- The shared resource merely represents the collection of items as a queue, and all the servers in the compute farm request items from the workload queue as they have resources available to complete the work.

Pull Workload Distribution

- The shared resource arbitrates between the many compute farm servers to ensure that each server has a chance to do work.
- If a compute farm server should for some reason fail, then it no longer requests additional workload items from the shared resource.
- The pull model for compute farms builds intelligent availability into the design of the cluster itself.
- No client exists that might request work from a failed server (as is the case in the load-balanced model).
- The servers themselves are the initiators of workload item requests.
- A failed server no longer gets additional work to do because it is no longer requesting more work.

BizTalk Compute Farms

- The simplest form of a BizTalk Server cluster is a compute farm.
- A collection of servers is organized into a single BizTalk Server Group, and every server in the Group points at the same BizTalk Shared Queue database.
- Most of these servers are configured to participate in workload item processing; these servers are called Processing Servers (more on these later).
- Other servers are only permitted to receive requests for additional workload items to be completed from outside (or inside) the system; these other servers are called Receiving Servers.
- Both types of servers are members of any given BizTalk Server Group, but each follows a slightly different clustering model.

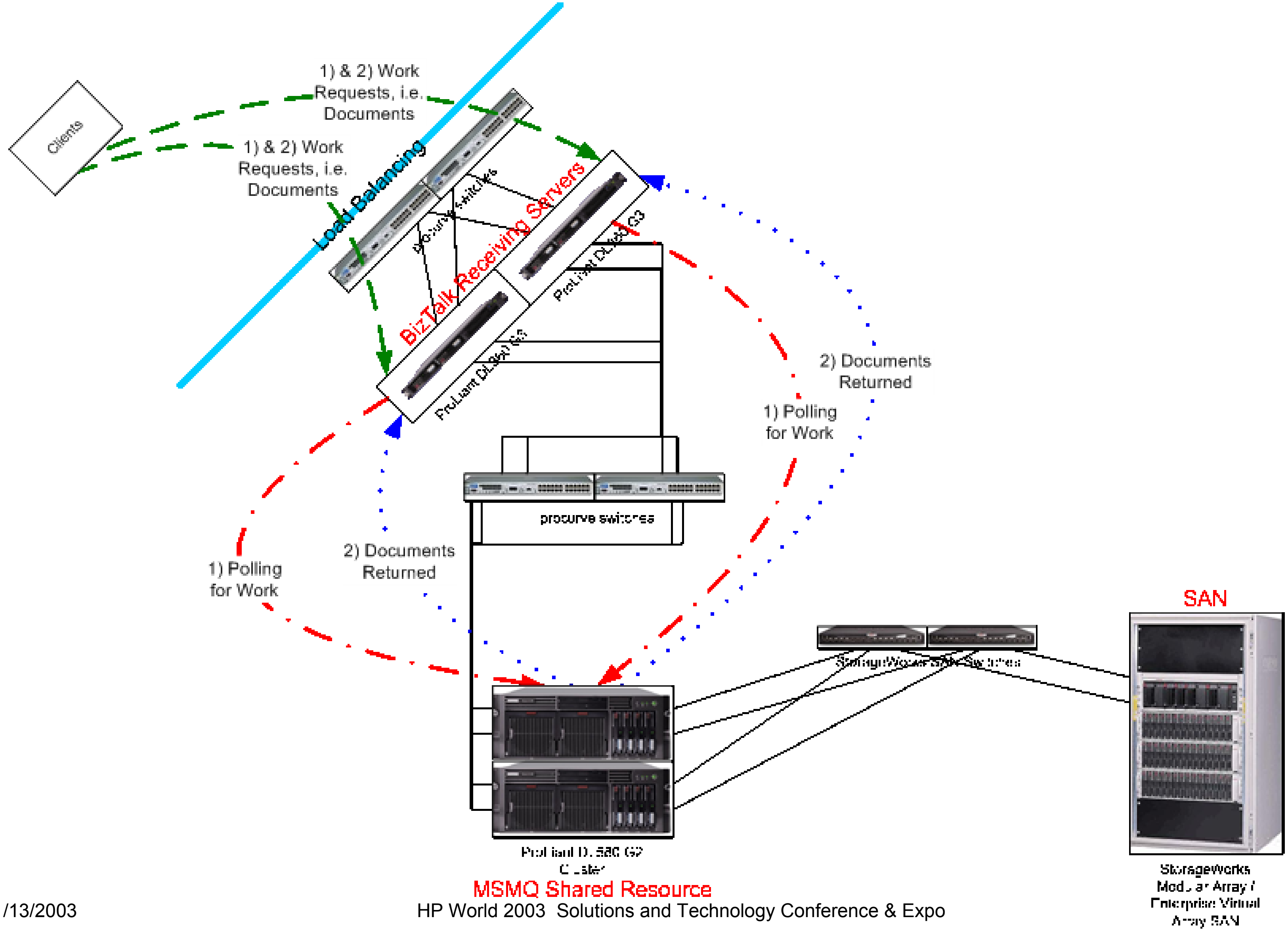
BizTalk Receiving Servers

- Listen for incoming workload item requests.
- Poll specific locations – such as a particular Message Queue – for those workload item requests.
- Workload items are called documents.
- Once the Receiving Servers receive a given document, they serialize it to a resource that is shared by all servers in the BizTalk Server Group.
- This shared resource is known as the BizTalk Shared Queue database, and it is physically housed on a database server or a traditional high availability database cluster.

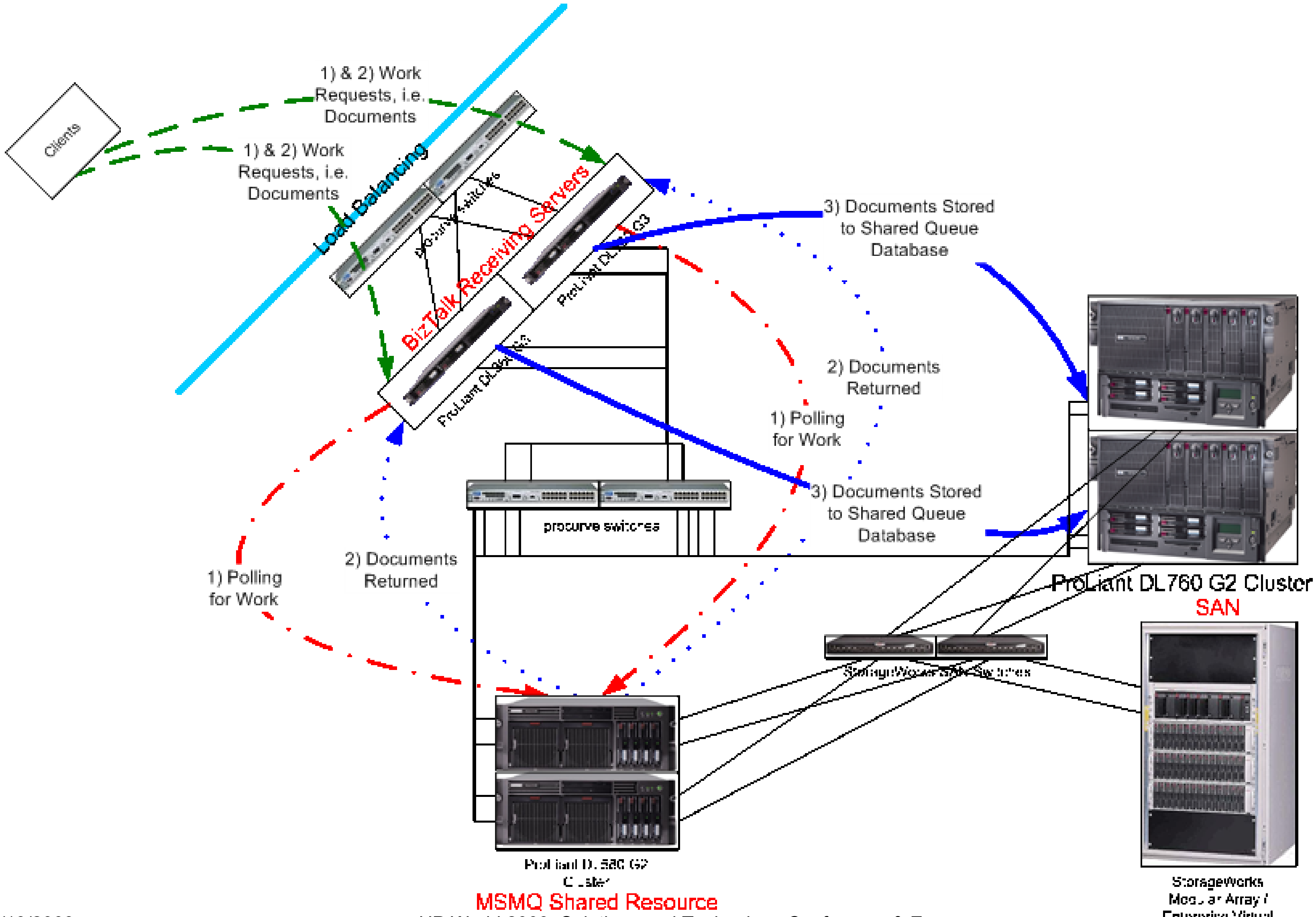
BizTalk Receiving Servers

- For listeners (e.g. HTTP listeners), load-balancing provides effective clustering using the push model for compute farms.
- For pollers, it's different:
  - Polling location (e.g. database table, file share, message queue) must be clustered for high availability.
  - Multiple Receiving Servers deployed.
  - Receiving Servers poll for workload items using the pull model.
  - High availability of the Receiving Servers is inherent to the design.

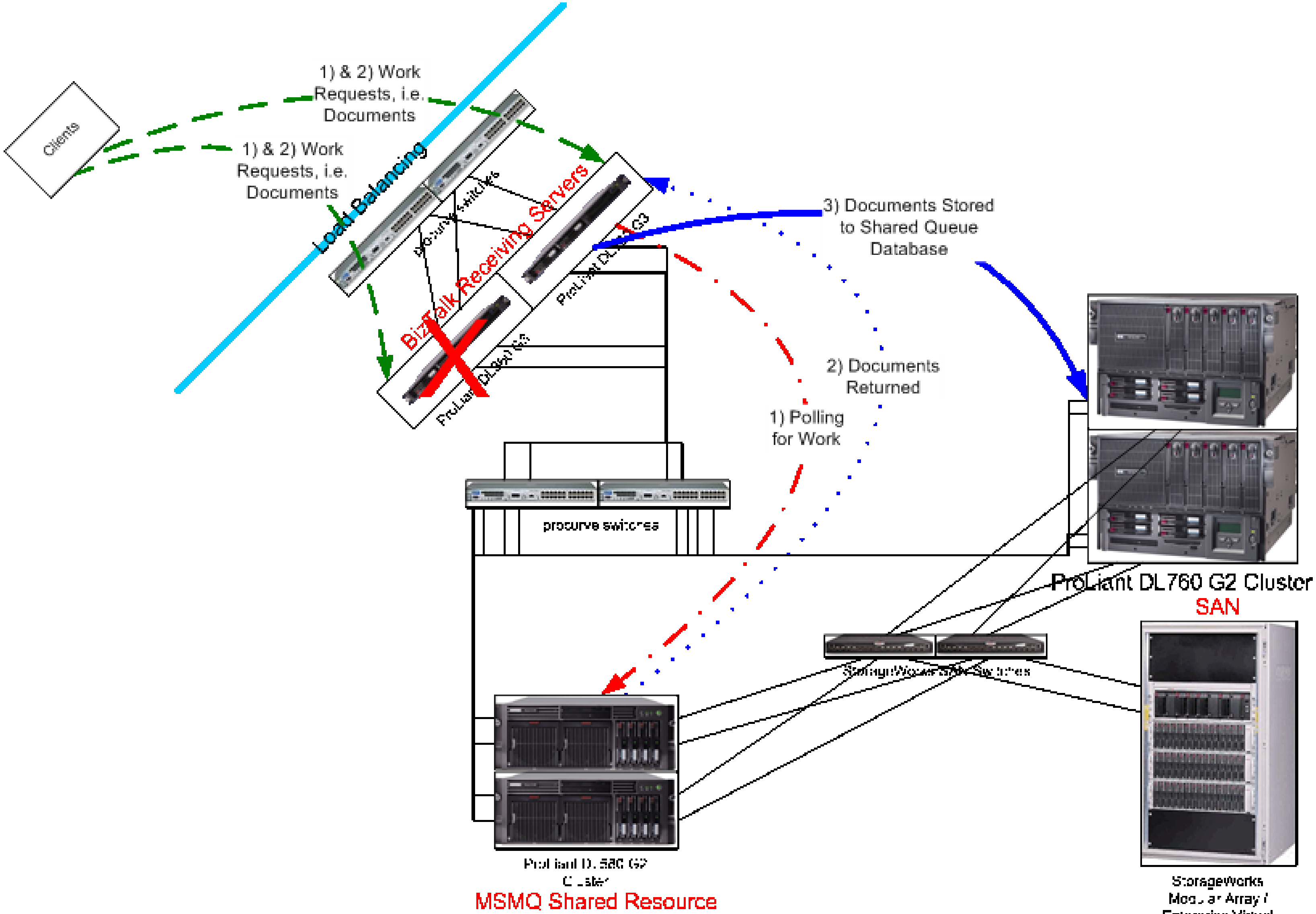
Receiving Servers: The Picture


Clients
1) & 2) Work Requests, i.e. Documents
1) & 2) Work Requests, i.e. Documents
Load Balancing
procurve switches
BizTalk Receiving Servers
ProLiant DL360 G3
ProLiant DL360 G3
2) Documents Returned
1) Polling for Work
1) Polling for Work
2) Documents Returned
procurve switches
StorageWorks SAN Switches
SAN
ProLiant DL580 G2 Cluster
MSMQ Shared Resource
StorageWorks Modular Array / Enterprise Virtual Array SAN

Workload Items Received and Stored


Clients
1) & 2) Work Requests, i.e. Documents
1) & 2) Work Requests, i.e. Documents
Load Balancing
procurve switches
BizTalk Receiving Servers
ProLiant DL380 G3
ProLiant DL380 G3
3) Documents Stored to Shared Queue Database
2) Documents Returned
1) Polling for Work
3) Documents Stored to Shared Queue Database
procurve switches
1) Polling for Work
2) Documents Returned
ProLiant DL760 G2 Cluster
SAN
StorageWorks SAN Switches
ProLiant DL580 G2 Cluster
MSMQ Shared Resource
StorageWorks Modular Array / Enterprise Virtual Array SAN

Workload Receipt with a System Failure

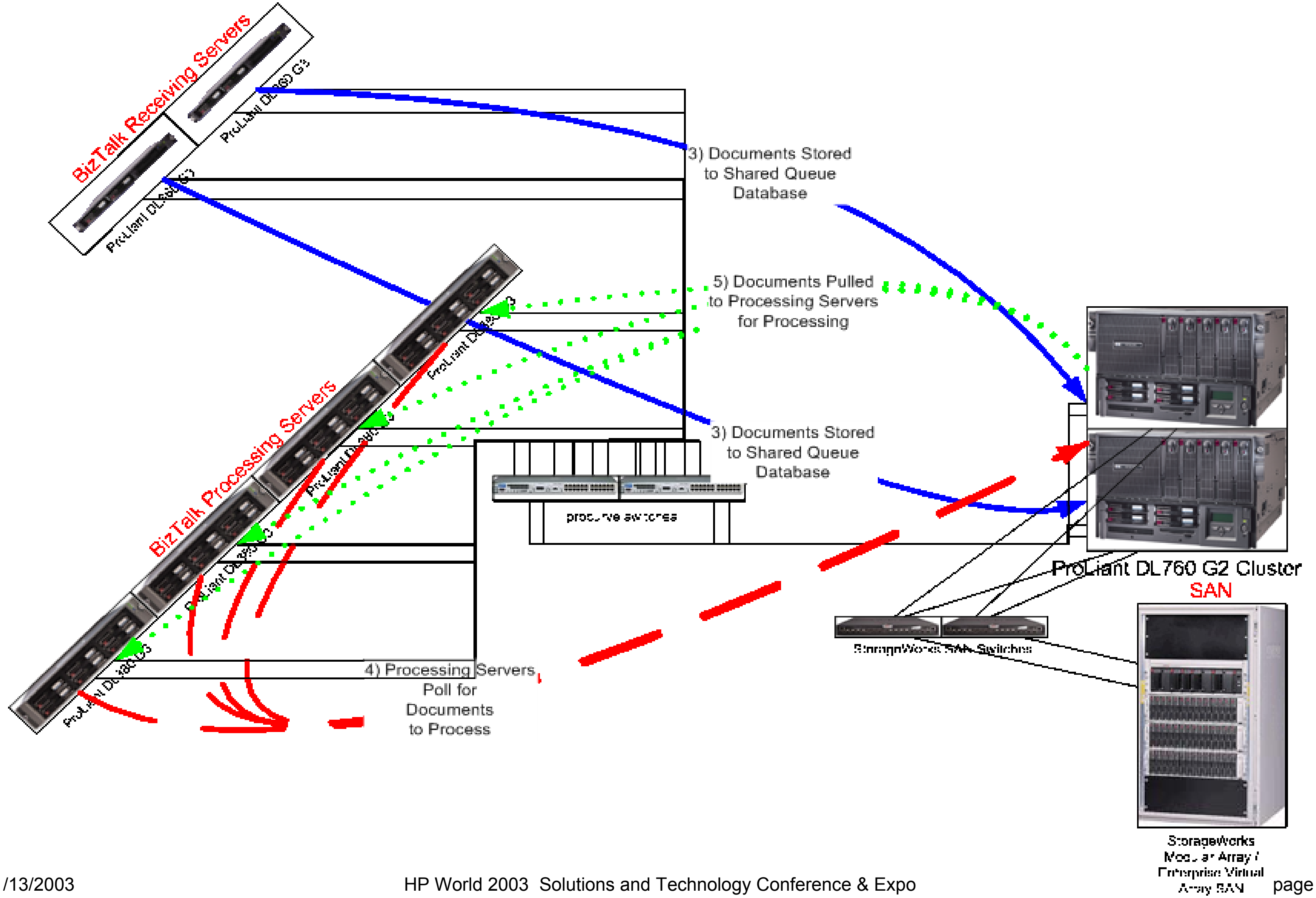


Clients
1) & 2) Work Requests, i.e. Documents
1) & 2) Work Requests, i.e. Documents
Load Balancing
procurve switches
BizTalk Receiving Servers
ProLiant DL360 G3
ProLiant DL360 G3
3) Documents Stored to Shared Queue Database
2) Documents Returned
1) Polling for Work
procurve switches
ProLiant DL760 G2 Cluster
SAN
StorageWorks SAN Switches
ProLiant DL580 G2 Cluster
MSMQ Shared Resource
StorageWorks Modular Array / Enterprise Virtual Array SAN

BizTalk Processing Servers

- As documents enter the Shared Queue database, BizTalk Processing Servers are polling for additional work that they can do.
- One Processing Server picks up a particular document in order to process it according to the business process defined by the organization which deployed it.
- Since each Processing Server can be configured to have several threads in its processing thread pool, several documents can be at various stages of processing on any particular Processing Server at a given point in time.
- As a Processing Server completes processing on its current documents, it requests additional items from the Shared Queue.

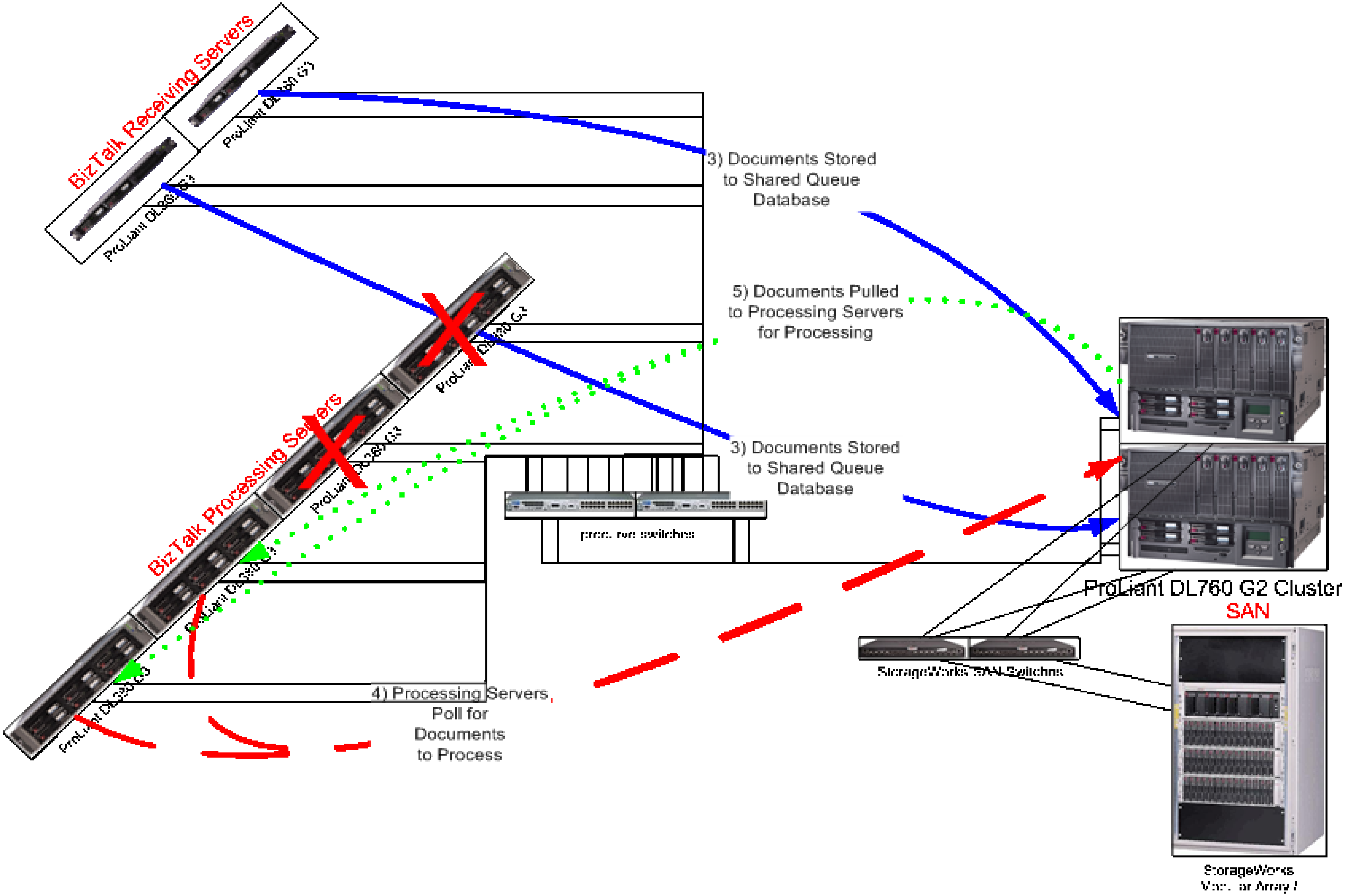
BizTalk Processing Servers: The Picture


BizTalk Receiving Servers
ProLiant DL360 G3
ProLiant DL360 G3
3) Documents Stored to Shared Queue Database
5) Documents Pulled to Processing Servers for Processing
BizTalk Processing Servers
ProLiant DL380 G3
ProLiant DL380 G3
ProLiant DL380 G3
ProLiant DL380 G3
ProLiant DL380 G3
3) Documents Stored to Shared Queue Database
ProCurve switches
ProLiant DL760 G2 Cluster
SAN
StorageWorks SAN Switches
4) Processing Servers Poll for Documents to Process
StorageWorks Modular Array / Enterprise Virtual Array SAN

BizTalk Processing Servers

- No load balancing is required in order to distribute the workload of documents from the Shared Queue database across all four BizTalk Processing Servers.
- The collection of BizTalk Processing Servers **always** behaves as a compute farm that uses a pull model to get workload items to each Processing Server.
- High availability is built into the system design.

BiTalk Processing Servers with Two System Failures


BizTalk Receiving Servers
ProLiant DL360 G3
ProLiant DL360 G3
BizTalk Processing Servers
ProLiant DL360 G3
ProLiant DL360 G3
ProLiant DL360 G3
ProLiant DL360 G3
proc. net switches
ProLiant DL760 G2 Cluster
SAN
StorageWorks SAN Switches
StorageWorks Virtual Array / Enterprise Virtual Array SAN

3) Documents Stored to Shared Queue Database

5) Documents Pulled to Processing Servers for Processing

3) Documents Stored to Shared Queue Database

4) Processing Servers Poll for Documents to Process

Complex BizTalk Clusters

- Until now, the discussion has pondered only the simpler cases of BizTalk Server deployments. These simpler cases are those which generally meet the following criteria:
  - Any BizTalk Server System that does **not** use BizTalk Orchestration.
  - Any BizTalk Server System that includes BizTalk Orchestration under the following circumstances:
    - Every BizTalk Server Orchestration Schedule in the system includes one, and only one, entry point.
    - No BizTalk Server Orchestration Schedule in the system invokes a synchronous COM (or other) call that takes more than a few seconds to return.

Complex BizTalk Clusters

- Some BizTalk Server deployments require the development and deployment of more complex business processes.
- Such business processes often include several long-running transactions which are best implemented using a BizTalk Server Orchestration Schedule.
- Such deployments require some additional attention to high availability within the system.

Orchestration Schedules with Multiple Entry Points

- The initial entry point starts the execution of the Orchestration schedule from the beginning.
- Schedule quickly moves from one step of the schedule to the next until the next entry point is hit.
- Execution must pause to wait for an external resource to invoke that entry point.
- This external resource is often an approval process or a process that runs in parallel to the earlier portions of the Orchestration schedule.

Multiple Entry Points Example

- Purchase order received by BizTalk Server and sent to an Orchestration schedule for processing.
- First several steps examine the total amount of the purchase order and log the purchase order to a database or ERP system.
- If the total amount is less than $500, the Orchestration schedule logic might assume that the purchase order is approved and forward it directly to the appropriate vendor.
- If the total is greater than $500, the schedule passes its flow of control to a second entry point into the schedule.
- A purchasing application would note that the Orchestration process has written a new purchase order to the database or ERP system.
- An approver examines these purchase orders once a day to approve or deny the order.
- Upon approval or denial, the purchasing application would re-invoke the Orchestration schedule at the point where it had paused, causing it to forward the purchase order to the appropriate vendor (if approved) or to log it to a denials database (if denied).

Multiple Entry Points Example

- Since the running schedule can exercise no control over when an external resource invokes a later entry point, it could be hours or even days before that entry point is called.
- If the running Orchestration schedule logged a new purchase order immediately after the daily approval / denial process, the schedule would have to pause until that process occurred again the following day.
- If the schedule remains in memory in its paused state, it is wasting resources on the server that executes the Orchestration.
- Thousands of paused schedules can easily consume a large amount of any server's resources, limiting the usefulness of that server.
- Failure of any server with thousands of Orchestration schedules in memory awaiting completion would mean the loss of too much valuable data.
  - Server failure would cost the loss of any uncompleted schedules.
  - It might be 1000 schedules worth $1000 each, or 1 schedule worth $100,000.
  - Either way its beaucoup bucks.

Problem Solved

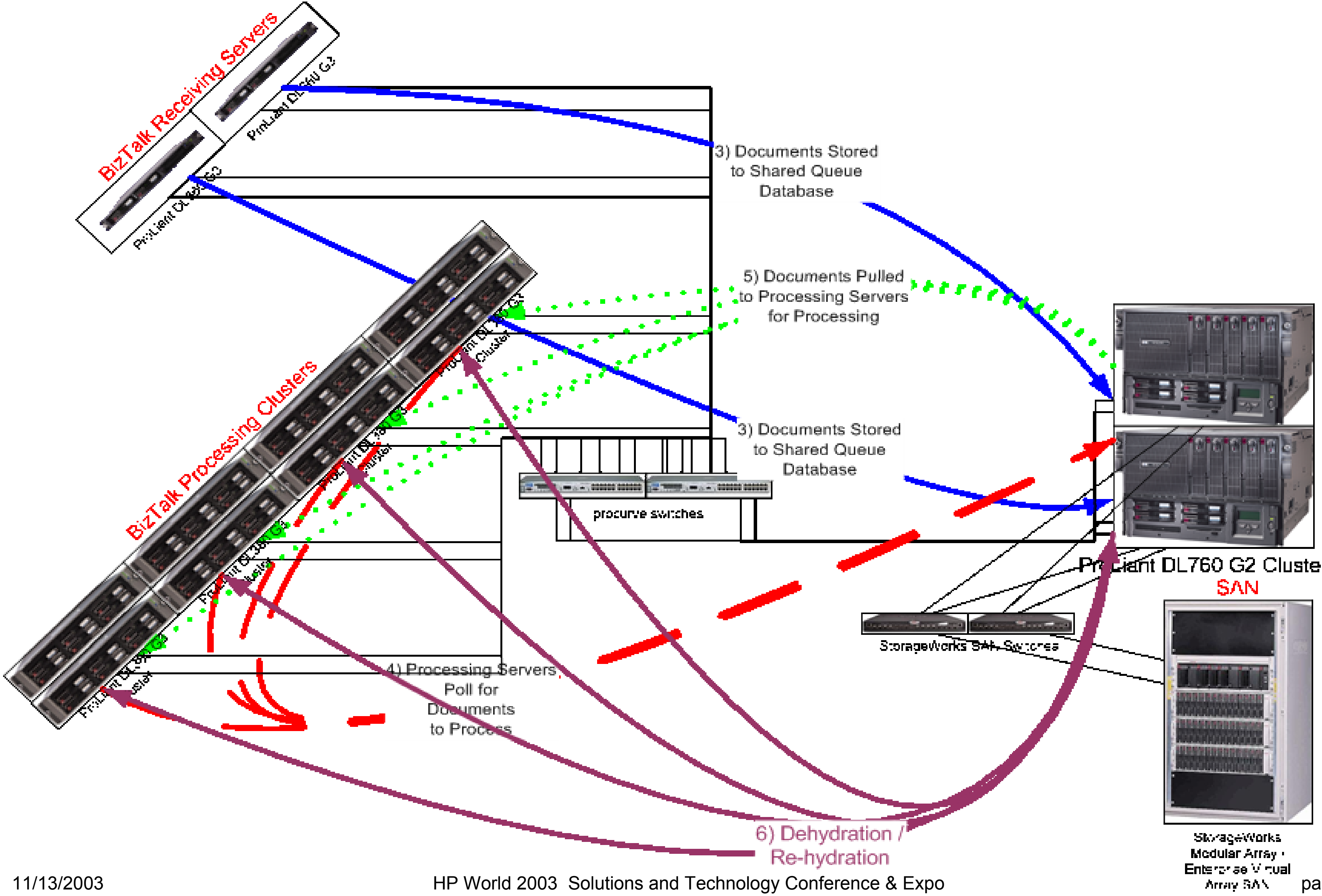
- BizTalk Server solves this problem by serializing such paused Orchestration schedules to a database after they have been paused in memory for a few seconds.
- Both the state of execution of the schedule and the data associated with the schedule are serialized.
  - This process is known as **dehydration**, and it is the means by which BizTalk Server protects running Orchestration schedules from loss in the event of a single server failure.
  - It is also the means by which BizTalk Server limits the amount of system resources that can be consumed by lots of paused schedules.
- Dehydrated schedules remain in what is known as the Orchestration Persistence database.
  - Once another entry point is invoked externally, the schedule is de-serialized from the Orchestration Persistence database.
  - Execution continues from the entry point in question to the next entry point into the schedule, or until the schedule completes execution.
  - This process is known as **re-hydration**.

New Problem Created

- Dehydrated Orchestration schedules have an affinity with the server name that dehydrated them.
- They cannot rehydrate on server with a different host name.
- They remain in the Orchestration Persistence database until the same host can re-hydrate them.
- What if that host fails?

- Must cluster Orchestration servers (i.e. Processing Servers) for high availability.

Problem REALLY Solved: The Picture



BizTalk Receiving Servers
ProLiant DL360 G3
ProLiant DL360 G3
BizTalk Processing Clusters
ProLiant DL 380 G3 Cluster
ProLiant DL 380 G3 Cluster
ProLiant DL 380 G3 Cluster
ProLiant DL 380 G3 Cluster
procurve switches
ProLiant DL760 G2 Cluster
SAN
StorageWorks SAN Switches
StorageWorks Modular Array · Enterprise Virtual Array SAN

3) Documents Stored to Shared Queue Database

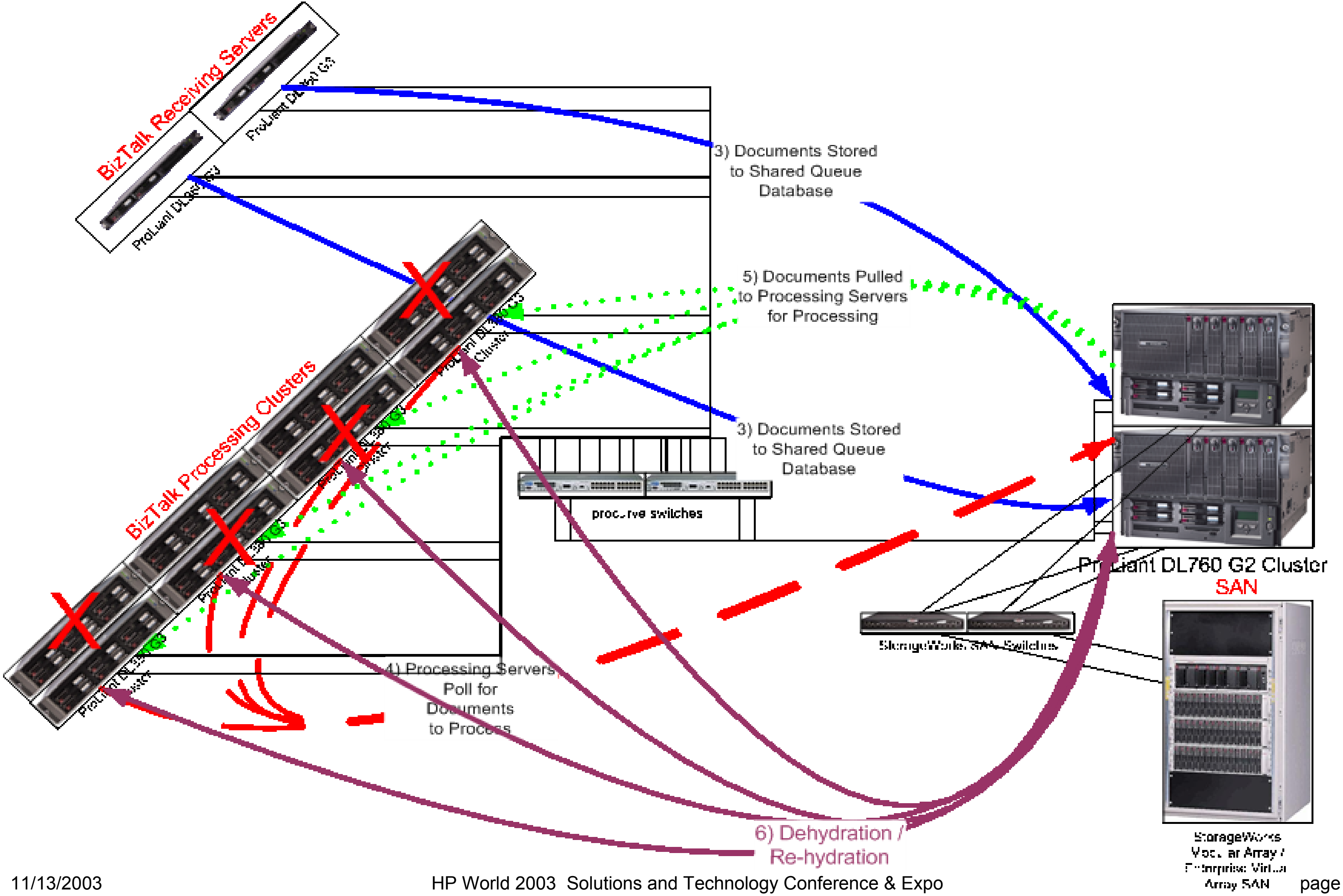
5) Documents Pulled to Processing Servers for Processing

3) Documents Stored to Shared Queue Database

4) Processing Servers Poll for Documents to Process

6) Dehydration / Re-hydration

Problem REALLY Solved: System Failure



BizTalk Receiving Servers
ProLiant DL380 G3
ProLiant DL380 G3
BizTalk Processing Clusters
ProLiant DL380 G3 Cluster
ProLiant DL380 G3 Cluster
ProLiant DL380 G3 Cluster
ProLiant DL380 G3 Cluster
procurve switches
ProLiant DL760 G2 Cluster
SAN
StorageWorks SAN Switches
StorageWorks Modular Array / Enterprise Virtual Array SAN

3) Documents Stored to Shared Queue Database

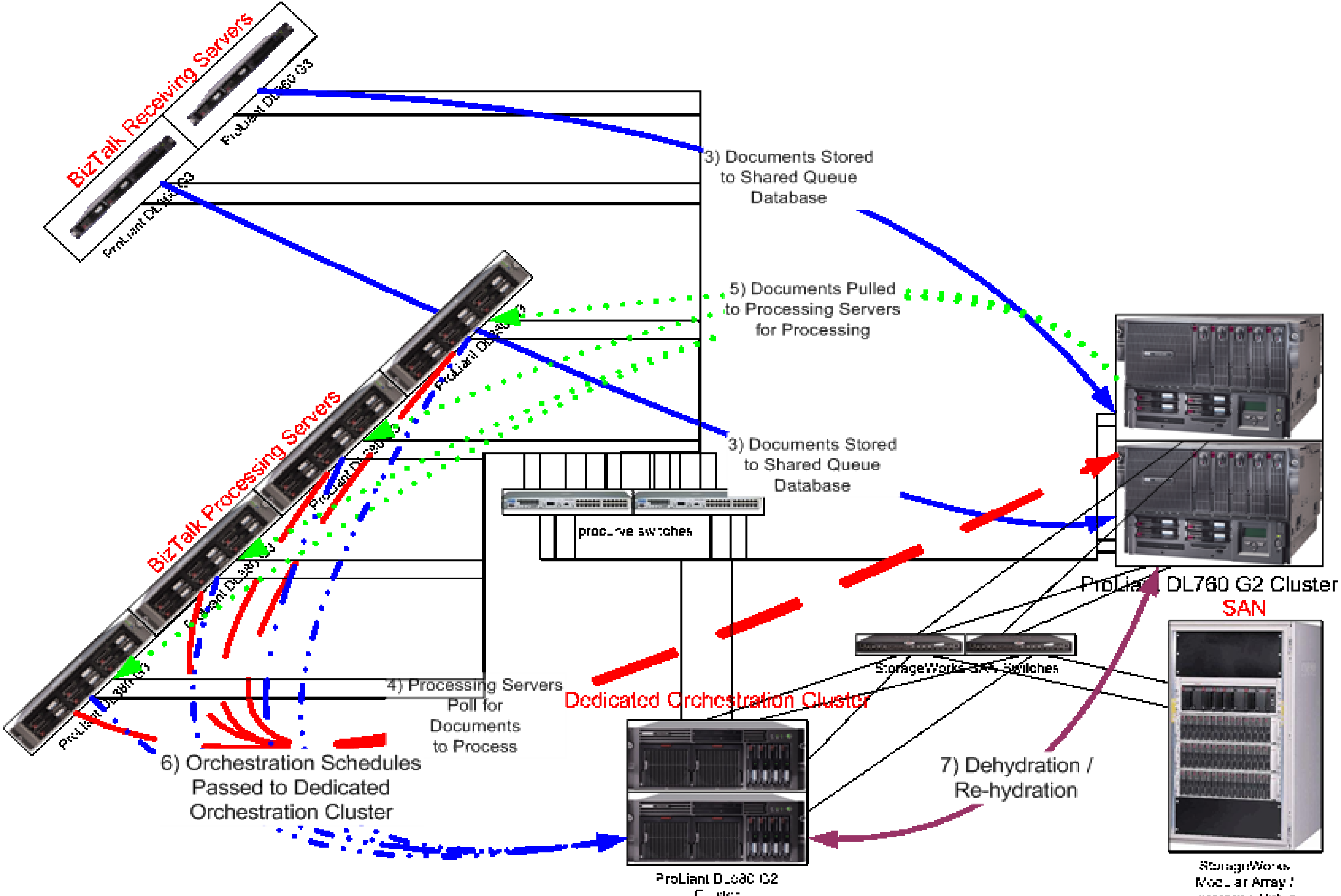
5) Documents Pulled to Processing Servers for Processing

3) Documents Stored to Shared Queue Database

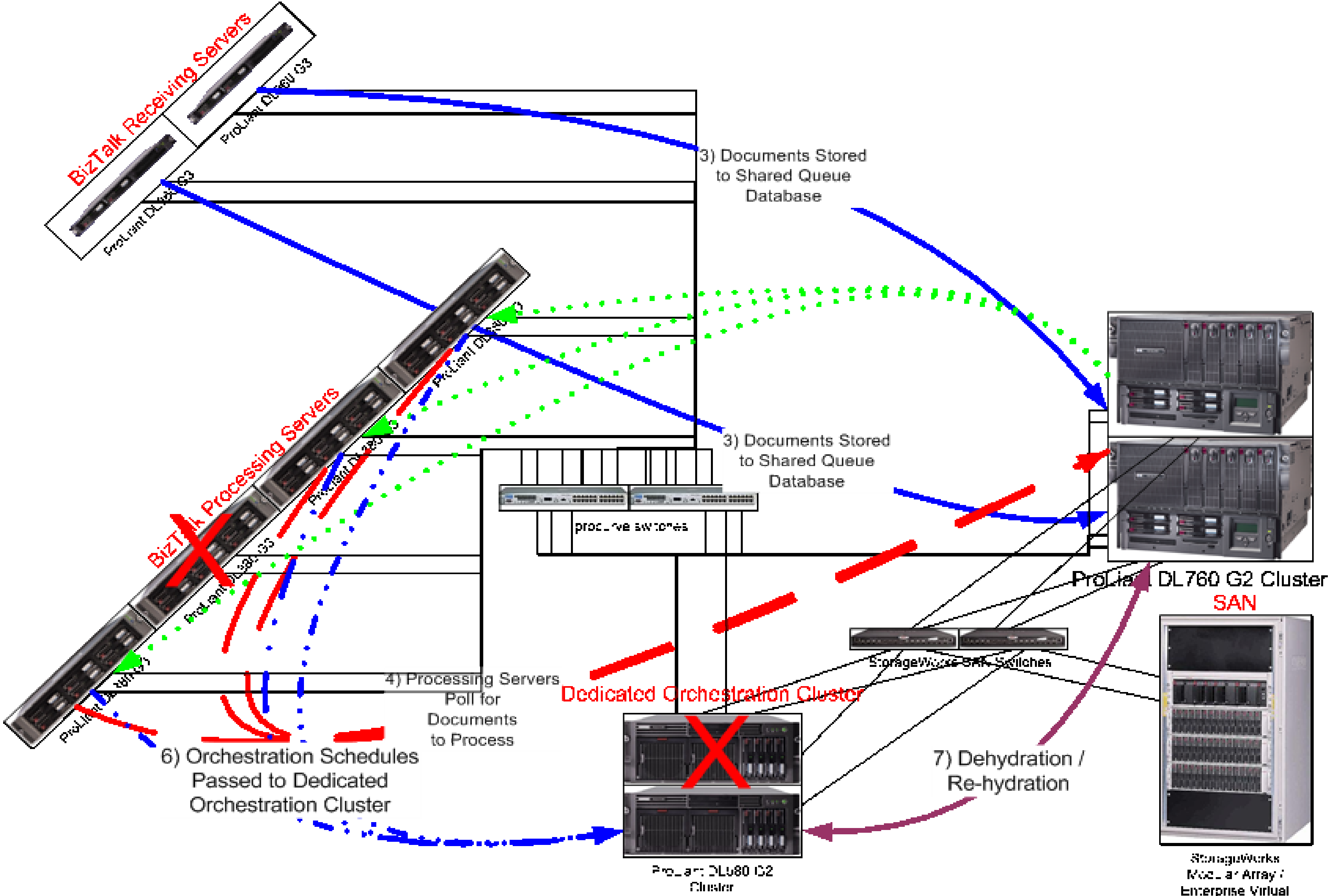
4) Processing Servers Poll for Documents to Process

6) Dehydration / Re-hydration

Problem REALLY Solved II


BizTalk Receiving Servers
ProLiant DL360 G3
ProLiant DL360 G3
3) Documents Stored to Shared Queue Database
5) Documents Pulled to Processing Servers for Processing
BizTalk Processing Servers
ProLiant DL380 G3
ProLiant DL380 G3
ProLiant DL380 G3
ProLiant DL380 G3
ProLiant DL380 G3
3) Documents Stored to Shared Queue Database
ProCurve switches
ProLiant DL760 G2 Cluster
SAN
StorageWorks SAN Switches
4) Processing Servers Poll for Documents to Process
Dedicated Orchestration Cluster
6) Orchestration Schedules Passed to Dedicated Orchestration Cluster
7) Dehydration / Re-hydration
ProLiant DL580 G2 Cluster
StorageWorks Modular Array / Enterprise Virtual Array SAN

Problem REALLY Solved II: System Failure



BizTalk Receiving Servers

ProLiant DL360 G3

ProLiant DL360 G3

3) Documents Stored to Shared Queue Database

3) Documents Stored to Shared Queue Database

BizTalk Processing Servers

ProLiant DL380 G3

ProLiant DL380 G3

ProLiant DL380 G3

ProLiant DL380 G3

ProLiant DL380 G3

procurve switches

ProLiant DL760 G2 Cluster

SAN

StorageWorks SAN Switches

4) Processing Servers Poll for Documents to Process

Dedicated Orchestration Cluster

6) Orchestration Schedules Passed to Dedicated Orchestration Cluster

7) Dehydration / Re-hydration

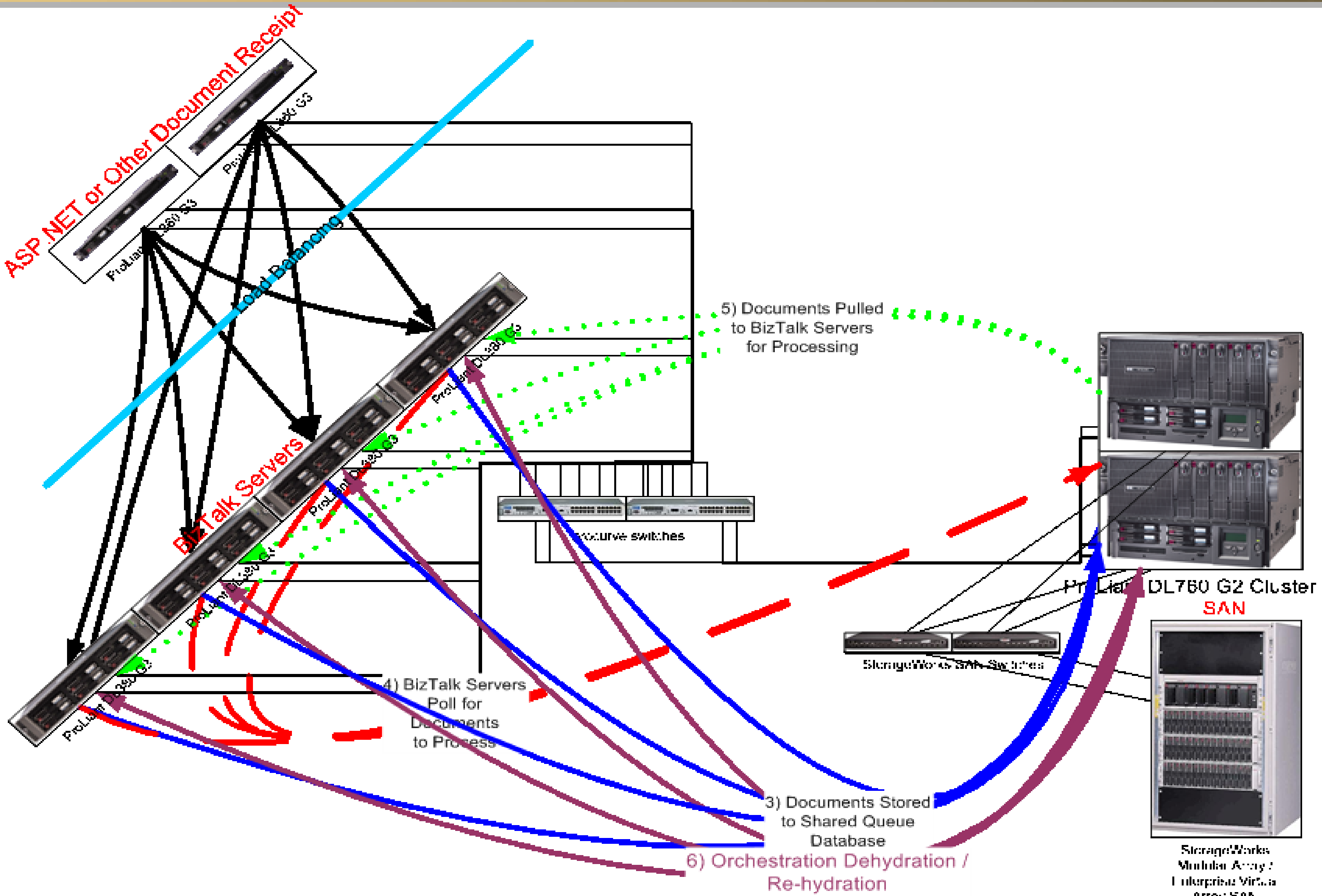
ProLiant DL580 G2 Cluster

StorageWorks Modular Array / Enterprise Virtual Array SAN

BizTalk 2004

- Architectural concepts are the same.
- Orchestration affinity to a specific host name is gone.
- Notion of Receiving Servers may be deprecated.
  - Servers can present Receive Locations using multiple protocols very easily.
  - Business processes can re-present themselves via multiple Receive Locations with little change.
  - Current beta seems to bolt documents directly to servers that received them for processing.

BizTalk 2004



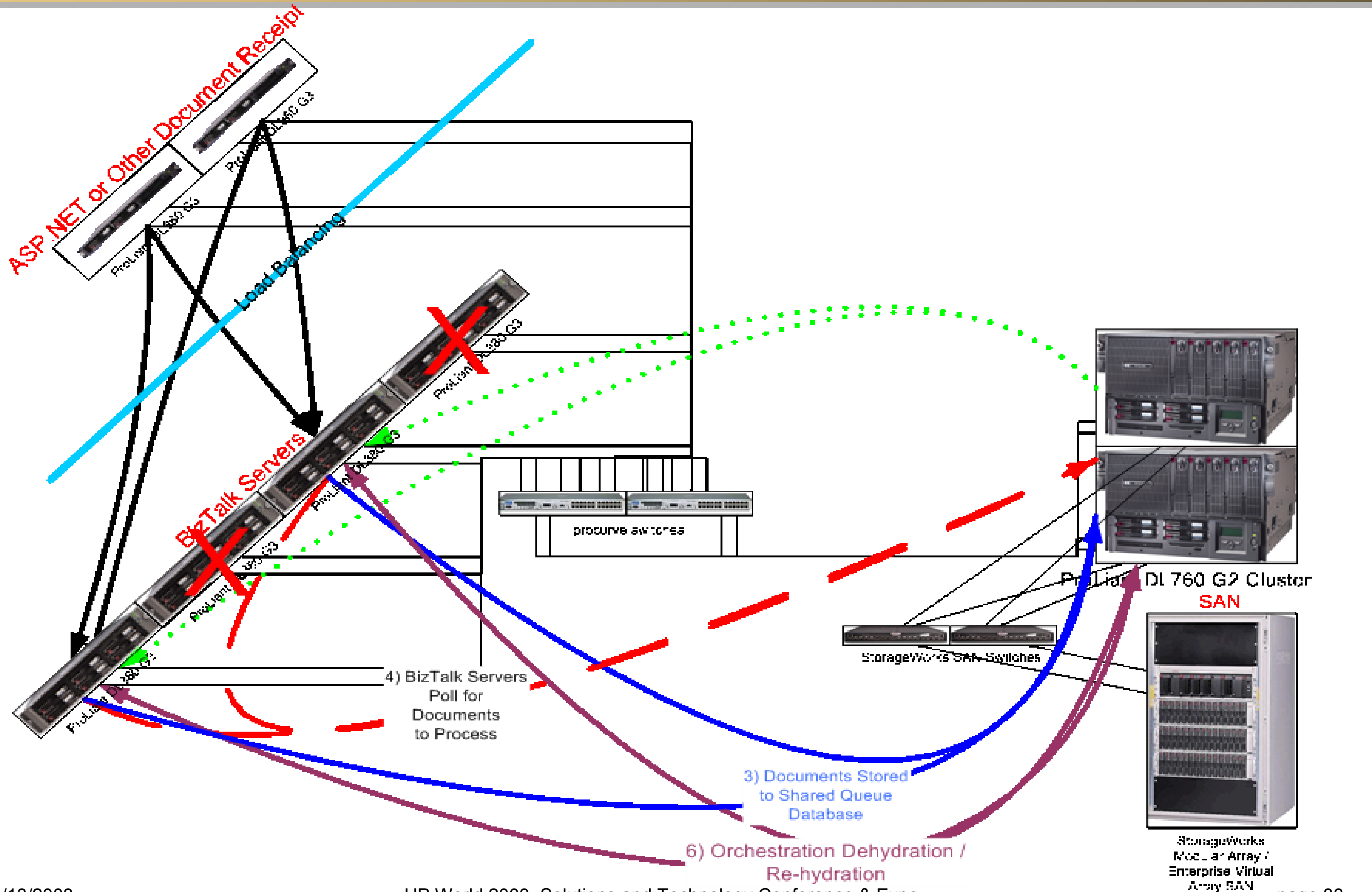
5) Documents Pulled to BizTalk Servers for Processing

4) BizTalk Servers Poll for Documents to Process

3) Documents Stored to Shared Queue Database

6) Orchestration Dehydration / Re-hydration

BizTalk 2004



ASP.NET or Other Document Receipt

Load Balancing

BizTalk Servers

3) Documents Stored to Shared Queue Database

4) BizTalk Servers Poll for Documents to Process

6) Orchestration Dehydration / Re-hydration

BizTalk 2004: MSMQT

- MSMQ dropped from the picture
- Message Queue submission is done to MSMQT
  - MSMQT supports a subset of MSMQ functionality.
  - No imposed limit on message or queue size.
  - Submission to MSMQT is picked up instantaneously by BizTalk.

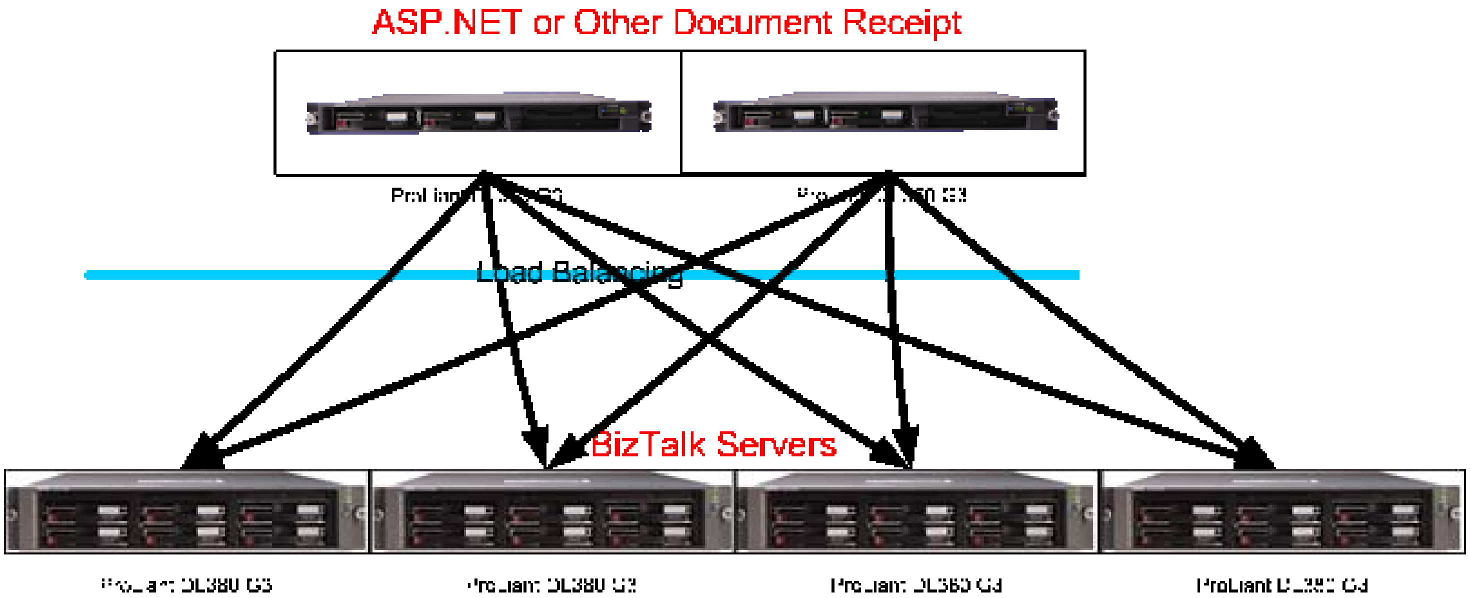
- **MSMQT allows BizTalk business processes to present themselves as if they were message queues.**

MSMQT

- Submitting server *must* have MSMQ installed.
  - Provides access to the MSMQ API implementations.
  - Still use System.Messaging or platform MSMQ APIs.
  - All submissions made to a remote queue.
- BizTalk server presents an MSMQT Receive Location as an input point to a business process.
  - Appears on the network as an MSMQ queue.
  - Received using the same API sets.
  - Can be load balanced using IP load balancing or NLB.
- BizTalk server must *not* have MSMQ itself installed.

MSMQT

- System.Messaging or Windows Platform SDK MSMQ APIs.
- MSMQ installed.



- MSMQ NOT installed.
- BizTalk business processes include MSMQT Receive Location.
- Entire business process can be presented over the network as if it were a remote MSMQ queue.

HP WORLD 2003

Solutions and Technology Conference & Expo

Interex, Encompass and HP bring you a powerful new HP World.

interex
shared knowledge • shared power

encompass
AN HP USER GROUP

hp
invent