Mission Critical Linux

http://www.mclx.com

kohari@mclx.com

High Availability Middleware For Telecommunications

September, 2002

Mission Critical Linux

- Founded in 1999 as an engineering company with financial backing from top name venture capitalist and private investors. We have raised over $26 million to date, funding all our technology initiatives
  - Highly skilled kernel, cluster, and network engineers from commercial UNIX backgrounds at Digital/Compaq/HP, IBM, and Sun

- Our focus is high availability middleware for the Linux OS

- We support all major Linux distributions across heterogeneous software and hardware platforms

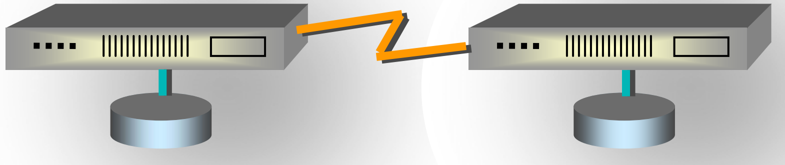
Modular Communications Platform

- **All Mission Critical Linux (MCLX) Software Supports Intel Hardware and Linux OS**
- **Strong Value Proposition for Developing Telecommunications Equipment using COTS (Commercially available Off The Shelf components)**
  - Development Costs, Support Costs and Time To Market considerations are more beneficial deploying on COTS vs. Proprietary (Research done by The Yankee Group)
- **MCLX software exports Service Availability Forum (http://www.saforum.org) recommended APIs Providing Maximum Application Portability**

HA Middleware: Two Primary Choices

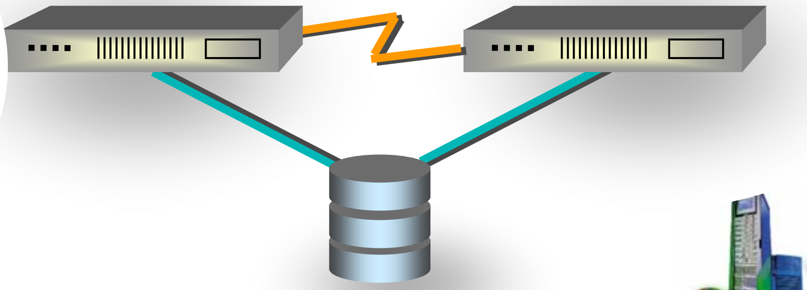
Shared Nothing Cluster

- Two independent servers
  - Each server connected/communicating via LAN
  - Each runs **CG Linux OS**, NFS, and HA Mgt SW
  - Each utilizes internal disk drives for the data store
  - Operates in Active/Active mode
- Pros
  - Can provide specified HA-NFS services easily
  - No external storage device to purchase or manage
  - Supports geographic separation of server pair
  - Can maintain locks, full NFS access perms during failover
  - Extremely fast failover speeds (40 milliseconds or less)
- Cons
  - Extra effort to maintain coherency w/ two data copies



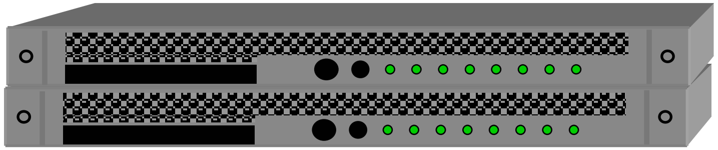
Shared Storage Cluster

- Two independent servers + external storage device
  - Each server connected/communicating via LAN
  - Each runs **CG Linux OS**, NFS, and HA Mgt SW
  - External storage device attached via FC, SCSI, or LAN and is used for the data store
  - Operates in Active/Active mode
- Pros
  - Can provide specified HA-NFS services easily
  - Easier management of single data copy
  - Literature supports
    - Can maintain locks, full NFS access perms during failovers
    - Write ordering/coherency is maintained at storage device
- Cons
  - Need to absorb expense of external storage device

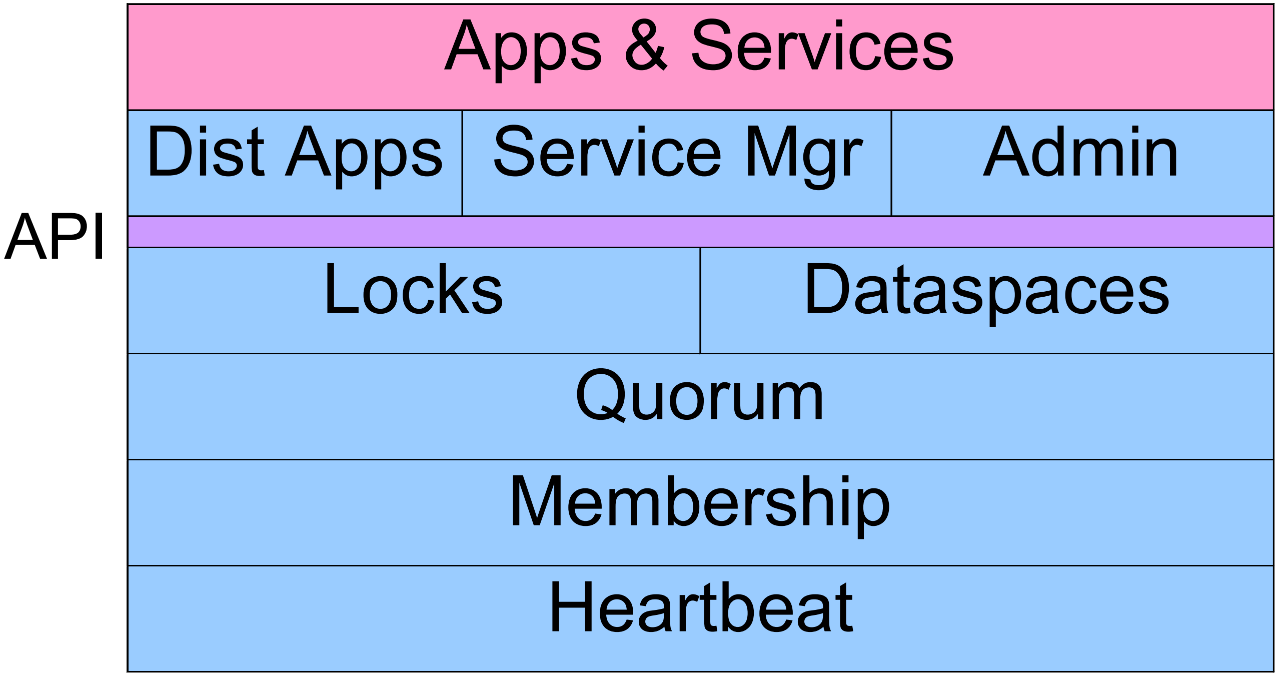


Shared Nothing Overview- NetGuard

- NetGuard has been rated as the top shared nothing cluster offering on Linux by a study published by D.H. Brown Associates (November 2001 – Real World Linux Clustering)
- Provides the highest levels of data integrity available on Shared Nothing clusters by integrating Distributed Raw Block Device (DRBD) and I/O barrier (power cycle capability). DRBD is the favored solution by Linux-ha.org
- Works equally well for both geographically distributed and local clusters
- Scales from 2 node clusters to 128 node clusters
- Provides for N+M clustering where any node in the cluster can act as the redundant node. Further, all nodes in the cluster are configured as active nodes (Active/Active)
- Fastest failover times available on Linux (< 40 millisecond failover)
- Provides for SNMP alerts.
- Open API for custom applications – Mission Critical Linux is a member of SA Forum.
- Built in load balancing support.
- NetGuard is capable of supporting all network file systems (such as NFS, CIFS, cluster file systems)



NetGuard Architecture



Apps & Services

Dist Apps | Service Mgr | Admin

API

Locks | Dataspaces

Quorum

Membership

Heartbeat

Heartbeat

- Transmit at configurable interval (.01 sec – 1.0 sec)
- Receives required in 3 intervals (configurable)
- Broadcast or Multicast packets
- Changes in membership reported to next layer

Membership

- Runs an agreement protocol
- Agreement is on a set of members
- Nodes see transitions in same order – prevents “split brain”
- Upon agreement, membership set reported to next layer

Quorum

- Majority of configured members required – prevents “split brain”
- Membership and quorum reported to apps that have registered

Distributed Data Service

- Provides in-memory dataspaces with optional persistent feature
- Open, close, read, write, notify
- Agreement protocol similar to membership
- Optimized reads
- App holds lock for read and write

Locking Service

- Distributed lock manager
- Open, close, lock, unlock

Service Manager

- A distributed application that uses the NetGuard* API
- Provides HA for unmodified applications and services

Distributed Applications

- Use the NetGuard* API (membership, locks, dataspaces)
- Can manage own availability, including hot standby – HA for state-full applications
- Can scale with node count

Cluster Administration

- Remote administration utility
- Manages cluster configuration
- Monitors service and member state
- Aids in bootstrapping
- Command-line interface for scripting
- Allows manual service relocation

Guarantees (arbitrary node failures)

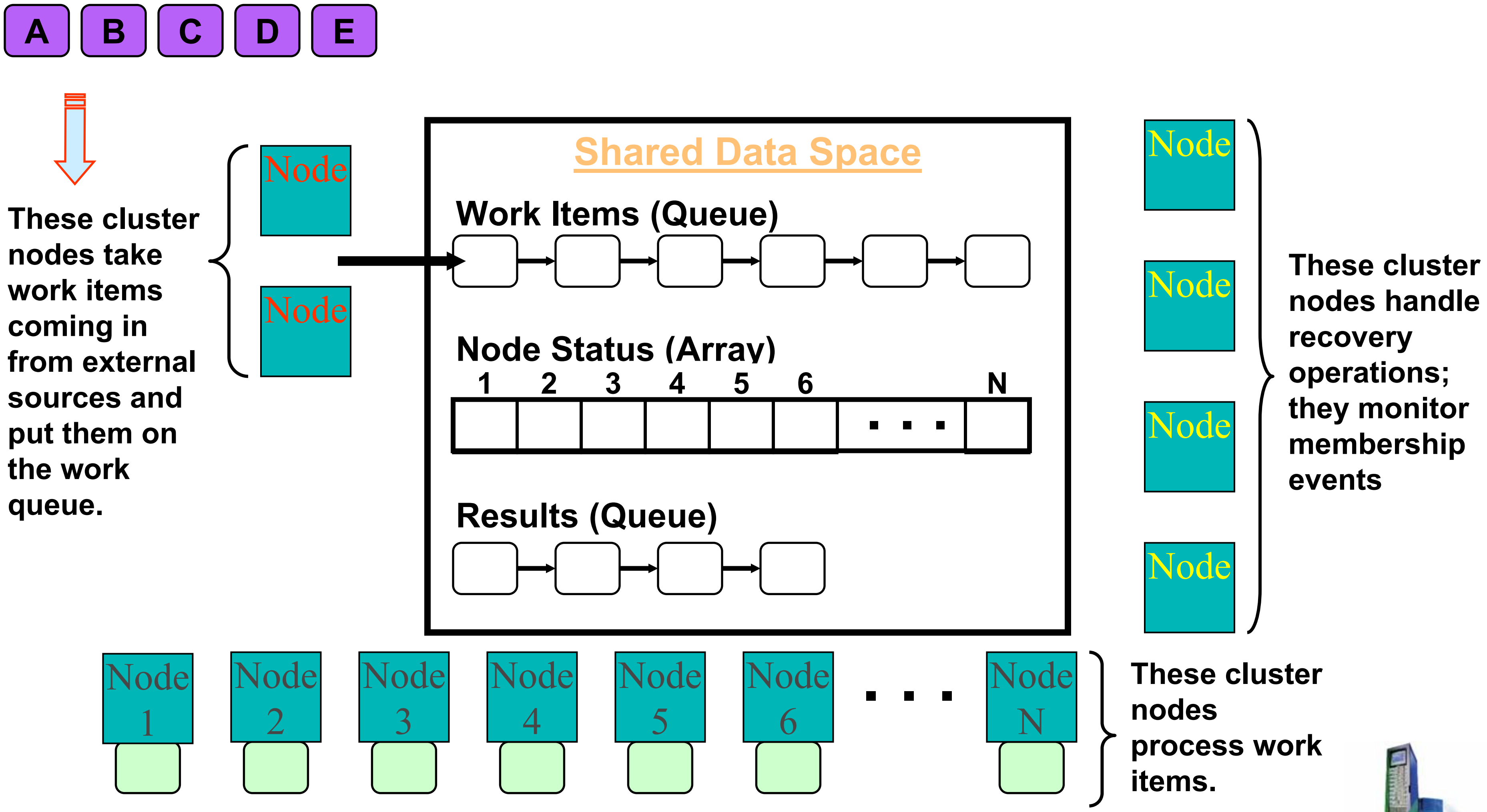
- Membership events are consistent and delivered in the same order on all nodes
- Locks always granted when free to one and only one node
- A lock waiter will eventually get the lock
- Ds_write is atomic: data is either written in its entirety or not at all
- Data written to a dataspace is visible on all nodes
- Ds_write notifications will be delivered
- Ds_read returns latest data

API: SMP To Cluster

	SMP	Cluster
Compute element	processor	node
Communication	memory	dataspace
Synchronization	locks	locks
Wait	condition wait	select
Wakeup	condition signal	ds write notif
HA	none	membership

Example: NetGuard* API

Shared Work Queue with Recovery



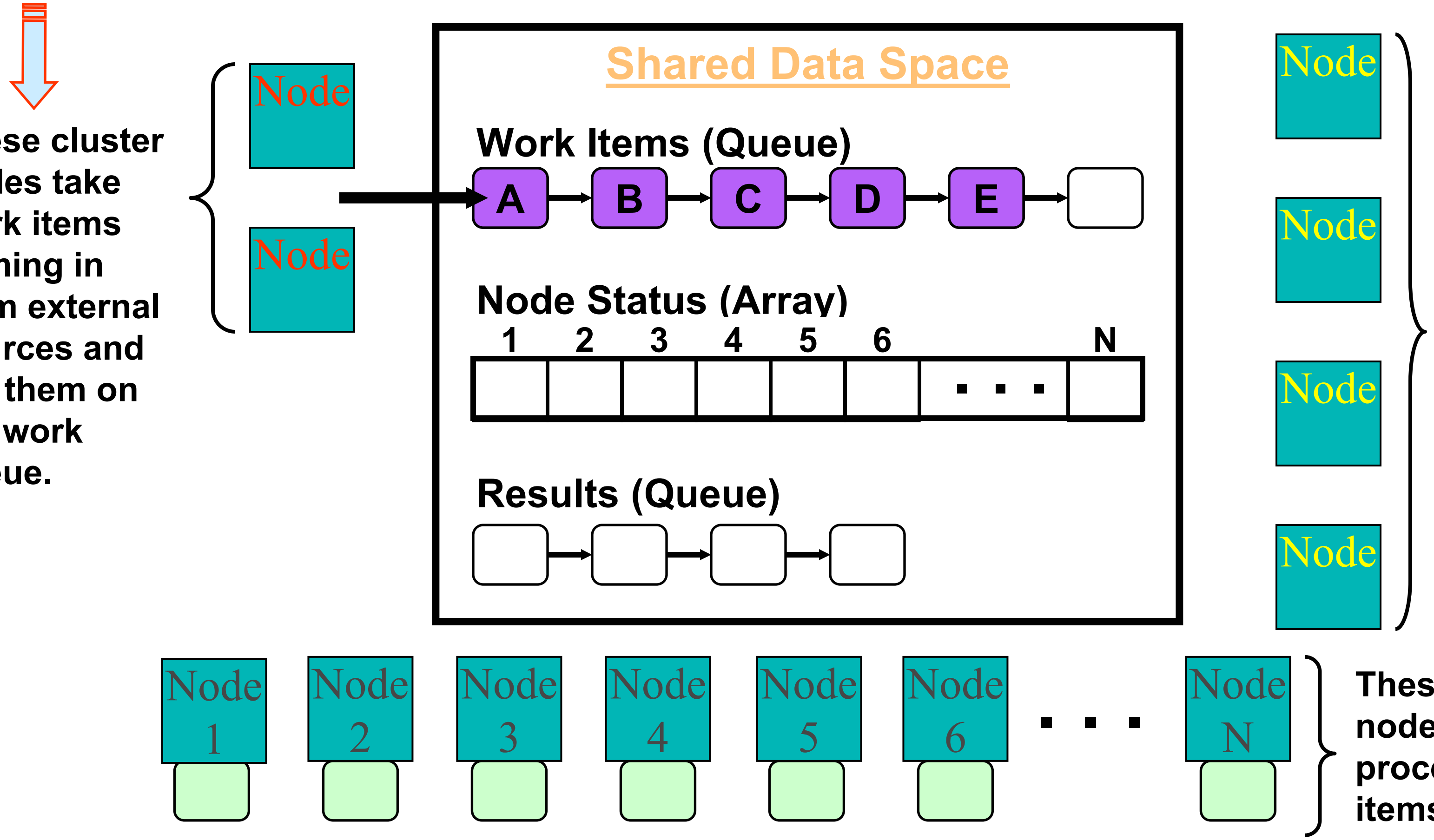
These cluster nodes take work items coming in from external sources and put them on the work queue.

These cluster nodes handle recovery operations; they monitor membership events

These cluster nodes process work items.

Example: NetGuard* API

Shared Work Queue with Recovery



These cluster nodes take work items coming in from external sources and put them on the work queue.

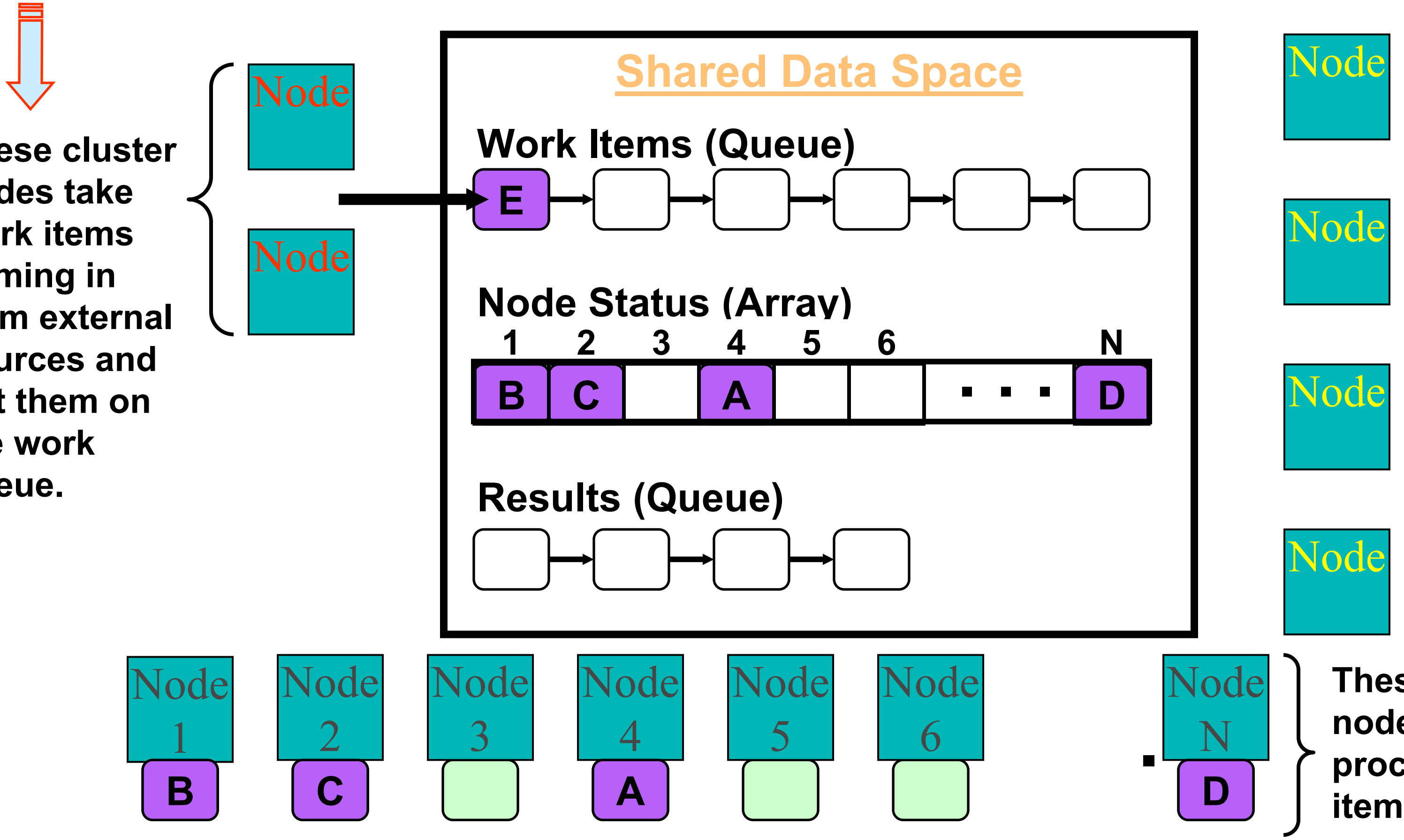
These cluster nodes handle recovery operations; they monitor membership events

These cluster nodes process work items.

Example: NetGuard* API

Shared Work Queue with Recovery

These cluster nodes take work items coming in from external sources and put them on the work queue.



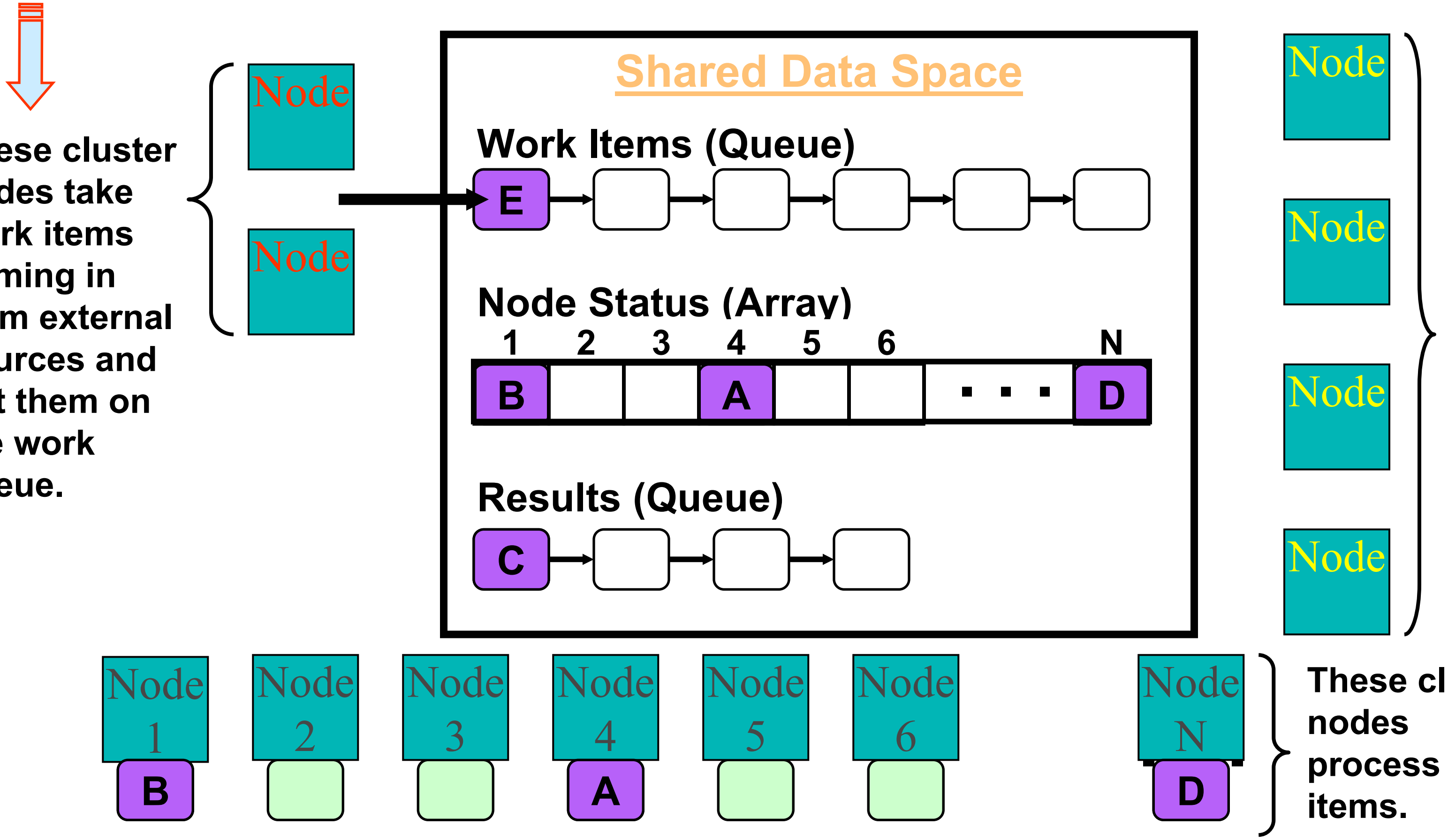
These cluster nodes handle recovery operations; they monitor membership events

These cluster nodes process work items.

Mission Critical Linux

Example: NetGuard* API

Shared Work Queue with Recovery



These cluster nodes take work items coming in from external sources and put them on the work queue.

Shared Data Space

Work Items (Queue)

Node Status (Array)

1	2	3	4	5	6		N
B			A			. . .	D

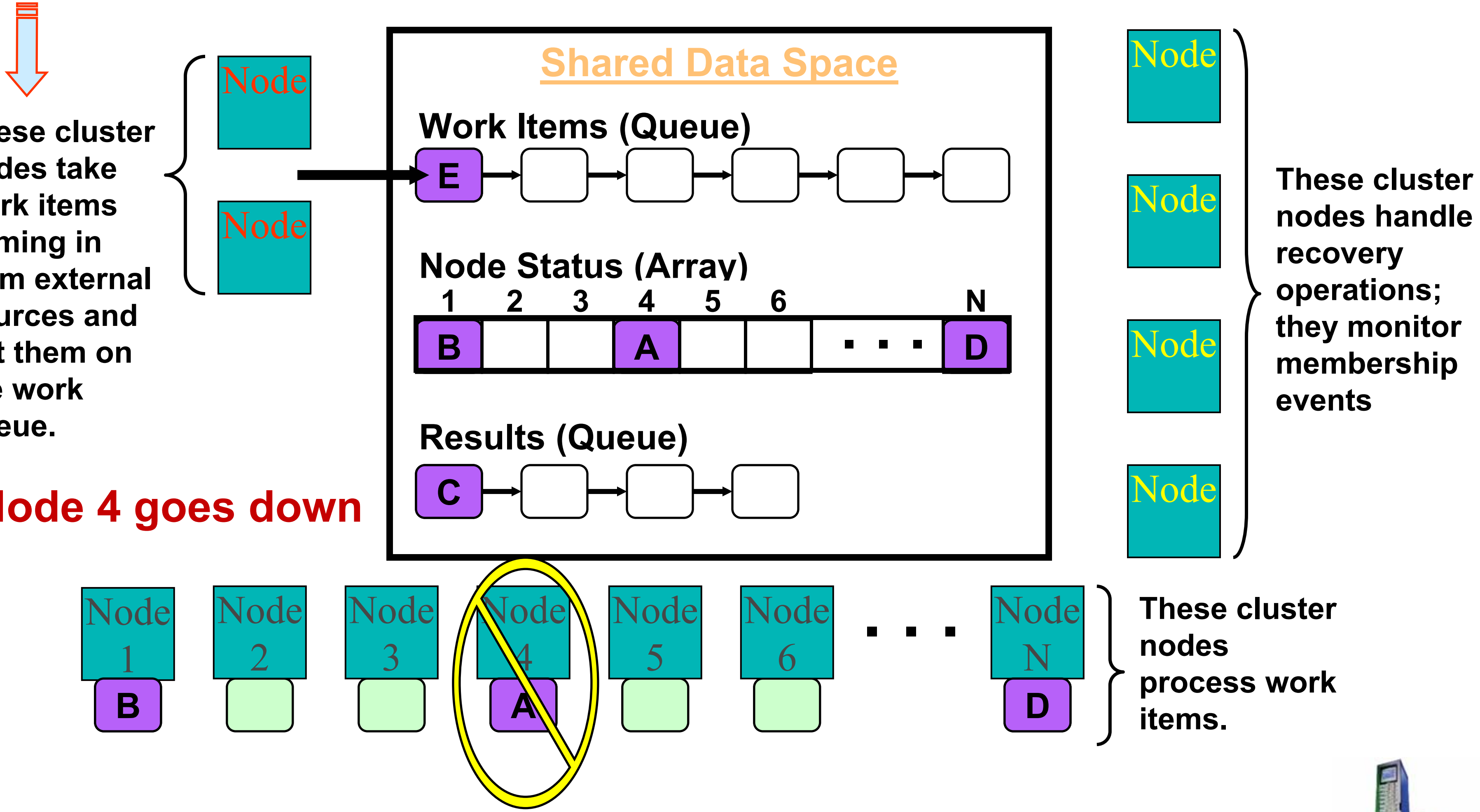
Results (Queue)

These cluster nodes handle recovery operations; they monitor membership events

These cluster nodes process work items.

Example: NetGuard* API

Shared Work Queue with Recovery



These cluster nodes take work items coming in from external sources and put them on the work queue.

Node 4 goes down

These cluster nodes handle recovery operations; they monitor membership events

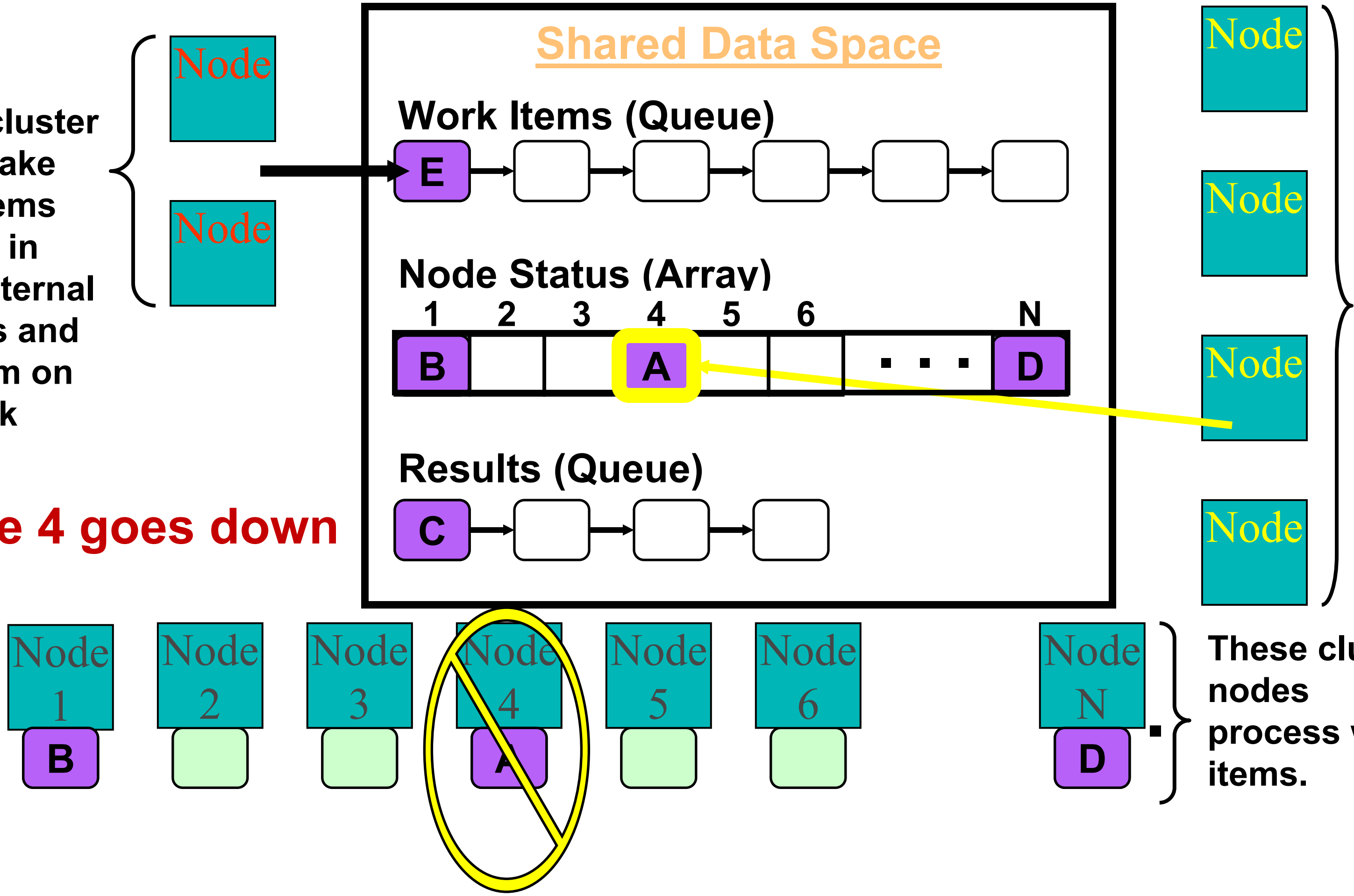
These cluster nodes process work items.

Example: NetGuard* API

Shared Work Queue with Recovery

These cluster nodes take work items coming in from external sources and put them on the work queue.

Node 4 goes down



Shared Data Space

Work Items (Queue)

E

Node Status (Array)

1	2	3	4	5	6	. . .	N
B			A				D

Results (Queue)

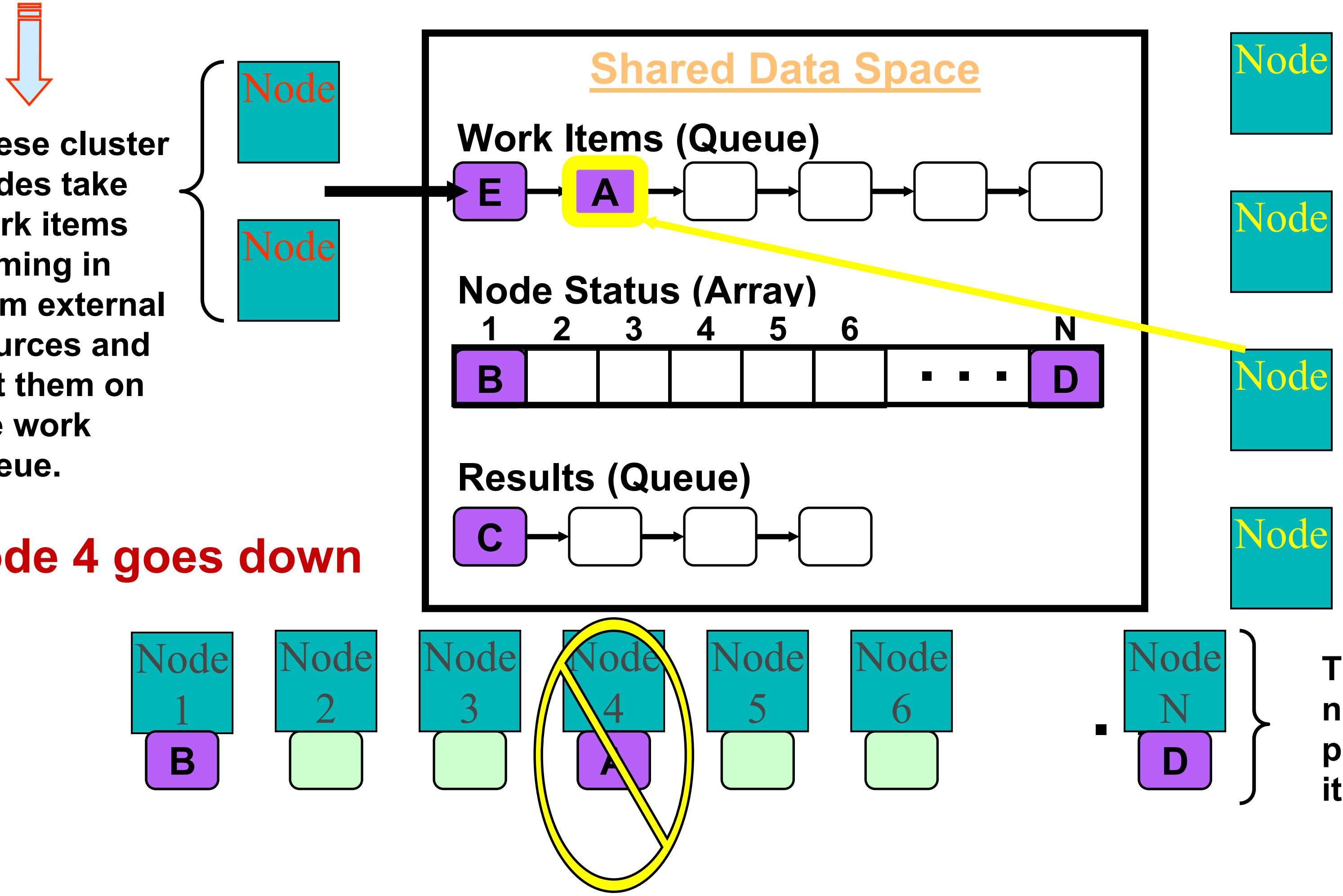
C

A recovery node finds that Node 4 is down and was processing work item “A”.

These cluster nodes process work items.

Example: NetGuard* API

Shared Work Queue with Recovery



These cluster nodes take work items coming in from external sources and put them on the work queue.

Work item "A" is moved back to the work queue so another node can process it.

Node 4 goes down

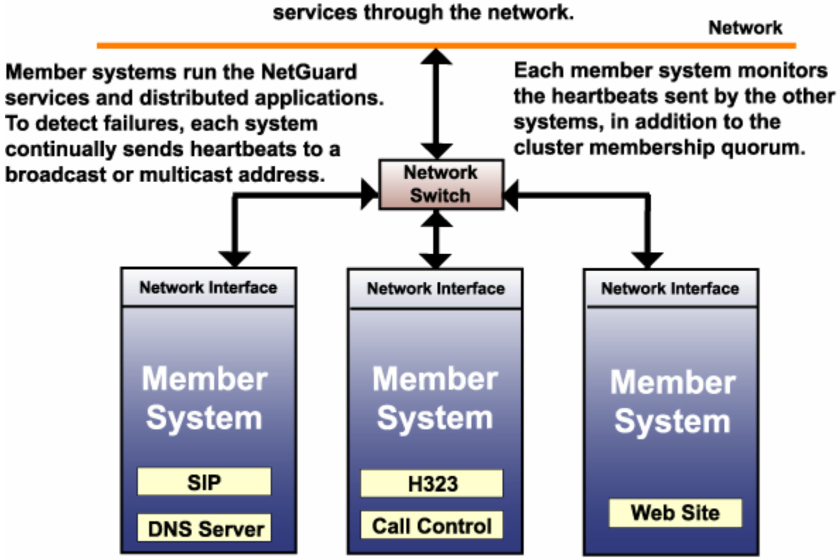
These cluster nodes process work items.

Mission Critical Linux

NetGuard* Services

NetGuard Cluster Operation

Clients communicate with the cluster services through the network.

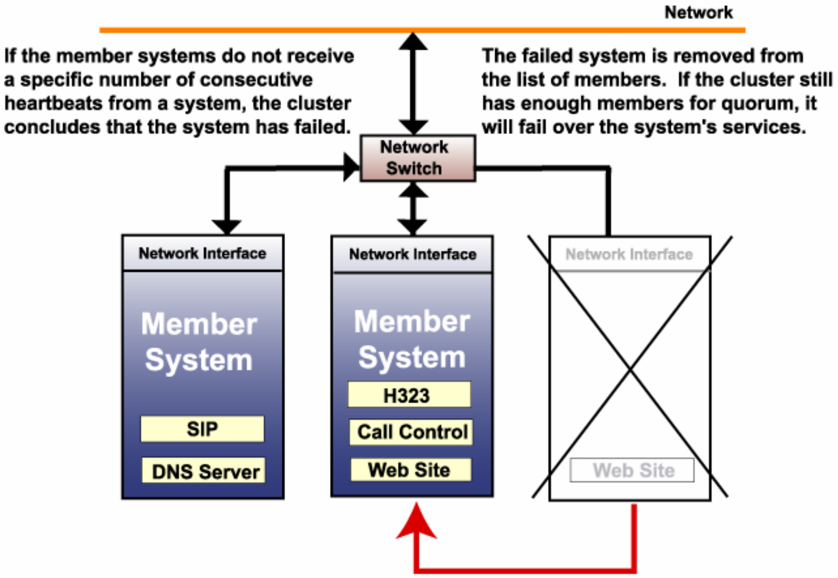


Member systems run the NetGuard services and distributed applications. To detect failures, each system continually sends heartbeats to a broadcast or multicast address.

Each member system monitors the heartbeats sent by the other systems, in addition to the cluster membership quorum.

Copyright 2001 Mission Critical Linux, Inc.

NetGuard Cluster Operation During Service Failover



If the member systems do not receive a specific number of consecutive heartbeats from a system, the cluster concludes that the system has failed.

The failed system is removed from the list of members. If the cluster still has enough members for quorum, it will fail over the system's services.

Copyright 2001 Mission Critical Linux, Inc.

Mission Critical Linux

http://www.mclx.com

kohari@mclx.com

High Availability Middleware For Telecommunications

September, 2002