hp software forum 2003

WHERE VISIONS CONNECT

OpenView Forum International

HP OpenView

WHERE VISIONS CONNECT

Session Number 1067

Mike Peckar

Fognet Consulting

The Ups and Downs of NNM Status Polling

Prepared by

Mike Peckar
Fognet Consulting

fognet.com

This session will focus on:

- Understanding NNM's default status polling behavior
  - How status polling has come to be "adaptive"
  - The entry points to customizing NNM's polling behaviors

- Alarms that convey status
  - Understanding their default behaviors and idiosyncrasies
  - How the number of alarms received are reduced by NNM

- Default Event Correlation logic that pertains to device status
  - Its effects on the alarms seen in the alarm browser
  - Its dynamic effects on status polling behavior
  - Everything you needed to know but were afraid to ask

In a nutshell

- *netmon* performs discovery and status polling
  - ICMP/IPX polls issued to *interfaces* discovered by NNM
  - SNMP status polls issued *to* agents, *about* interfaces
  - Polling intervals are set in different ways and dynamically
- The *event subsystem* conveys status to Alarm Browser & maps
  - Alarm browser is "node centric" through NNM V6.31
  - Map status is propagated from interfaces
- *Event Correlation* introduced in NNM V6.0
  - Connector Down correlation addressed cascade failures
  - Pair-Wise, Repeated Event correlation reduce status alarms
  - NodeIf correlation added in NNM V6.31 – tectonic changes
  - Status correlations tied heavily to *netmon*, event subsystem

NNM status through the ages

- NNM V5.0
  - Bridge MIB and unnumbered interface discovery & polling
- NNM V6.0
  - Connector down ECS & critical path analysis in *netmon*
  - Relational event display & status polling reduction.
- NNM V6.2
  - SNMP-based polling and object-based polling
  - Change default status settings to quiet unconnected ports
- NNM V6.31
  - Major change to default status event behaviors for Node/If
  - NodeIf, IntermittentStatus, status in Dynamic Views
- NNM V6.4
  - Status polling and ECS interval tweaks to reduce events

Summary of default status polling behavior

- Global default for any discovered interface: 15 minutes
  - Same for Layer 2 SNMP-discovered interfaces
  - Polls are scheduled by *netmon* and spread randomly to reduce load
- Status polling reduction for secondary failures
  - Interval doubles for interfaces beyond primary IF (V6.0+)
- Object-based polling (V6.2+)
  - Routers, bridges, hubs polled more frequently, Primary IF's even more
  - V6.4: poll ***less*** frequently (except primary Ifs); add node objects
- Dynamically-adjusting polling by *netmon* (V6.31+)
  - All down interfaces re-polled at 2 and 4 minutes
  - All up interfaces re-polled at 2 minutes
  - All connector interfaces immediately polled when one goes down

Layer 2 polling default status behaviors

- Support for Bridge, MAU, Repeater MIB; VLANs
- Un-numbered Ifs: inferred from port table, polled via ARP
  - Status (V5-V6.1): Critical/Normal; (V6.2+): Unknown/Normal
- SNMP status mapping fixed from V5 until V6.2 (all log-only)
  - Status reflected in maps, but not in Alarm Browser

ifAdminStatus	ifOperStatus	OV Status
down	any	DISABLED
testing	any	TESTING
up	up	NORMAL
up	down	CRITICAL
up	testing	TESTING

Layer 2 status configurable in NNM V6.2+

- *netmon.statusMapping* defines customizable SNMP status levels
- File contains colon separated triplets. Possible values:

ifAdminStatus	:	ifOperStatus	:	Status
up		up		unset
down		down		unknown
testing		testing		normal, up
any		unknown		critical, down
		dormant		disabled
		notpresent		unmanaged
		lowerlayerdown		restricted
		any		testing

SNMP-based polling NNM V6.2+

- *netmon.snmpStatus* – Status poll via SNMP by IP Addr. ranges
  - Designed to provide alternative to ICMP as status mechanism
  - Queries ifIndex, ifOperStatus, ifAdminStatus
  - Interface status set per *netmon.statusMapping* rules
  - Be careful to list only SNMP-supported devices
- netmon –k snmpTimeoutImplies=status   (lrf setting)
  - Possible values: unknown, unchanged, critical (default)
- Example $OV_CONF/*netmon.snmpStatus* file entries:

```
10.2.112.86 # tomcat
10.2.1-255.0-49
10.2.4-5.*
*.*.*.*
```

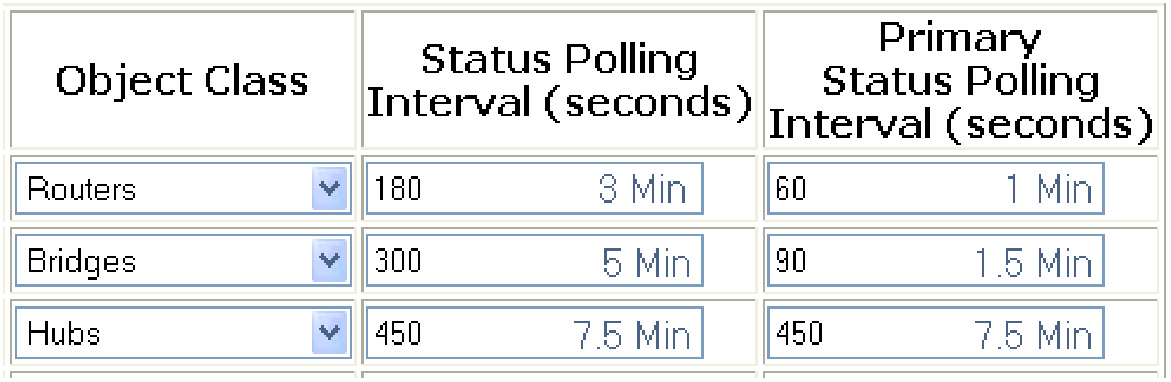
Object-based polling NNM V6.2+

- Options -> SNMP Configuration -> Poll Objects button
  - JAVA GUI to configure *netmon.statusIntervals*
  - Configuration file format: filter:interval:[primary if interval]
- Affects Default polling intervals out of box:
  - Tightens default polling intervals for Routers, Bridges, Hubs
  - ***Loosens*** default polling intervals for Nodes to 1h (V6.4)
- Uses *netmon's* Critical Path Analysis to determine primary
- Uses NNM standard filters and filter definition language
- Command line to determine polling interval for device/interface:

  xnmsnmpconf –resolve *target*
  nmdemandpoll –i *target* (issues polls)

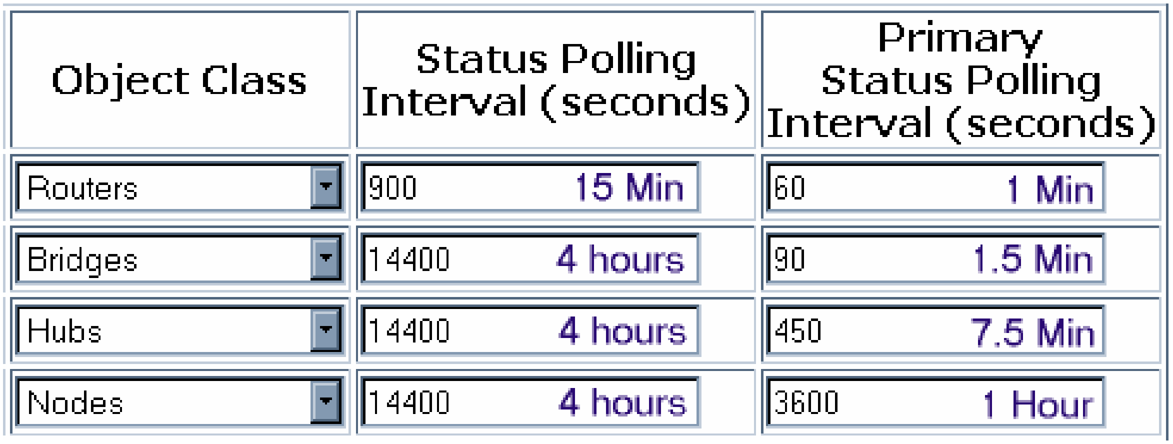
Object-based polling NNM V6.2+

- Default Object-based polling intervals:
  - NNM V6.2:
  - NNM V6.31:



Object Class	Status Polling Interval (seconds)	Primary Status Polling Interval (seconds)
Routers	180 3 Min	60 1 Min
Bridges	300 5 Min	90 1.5 Min
Hubs	450 7.5 Min	450 7.5 Min

  - NNM V6.4:



Object Class	Status Polling Interval (seconds)	Primary Status Polling Interval (seconds)
Routers	900 15 Min	60 1 Min
Bridges	14400 4 hours	90 1.5 Min
Hubs	14400 4 hours	450 7.5 Min
Nodes	14400 4 hours	3600 1 Hour

Dynamically-adjusting status polling

- V6.0 *netmon* enhancement to support Connector Down
  - Reduce status polls issued to secondary failure-mode If's
- V6.31 *netmon* status polling enhancements:
  - Priority status polling for multi-homed nodes
    - Old way: Stick to schedule for polling other IF's when one IF fails
    - New way: Immediately poll other "up" IFs on node; IFNode applies
  - False node down's
    - Old way: Stick to schedule for polling newly down interface
    - New way: Poll down IF at 2 minutes & 4 minutes; pair-wise applies
  - Flapping status
    - Old way: Stick to schedule for polling newly up interface
    - New way: Poll IF at 2 minutes; intermittentStatus circuit applies

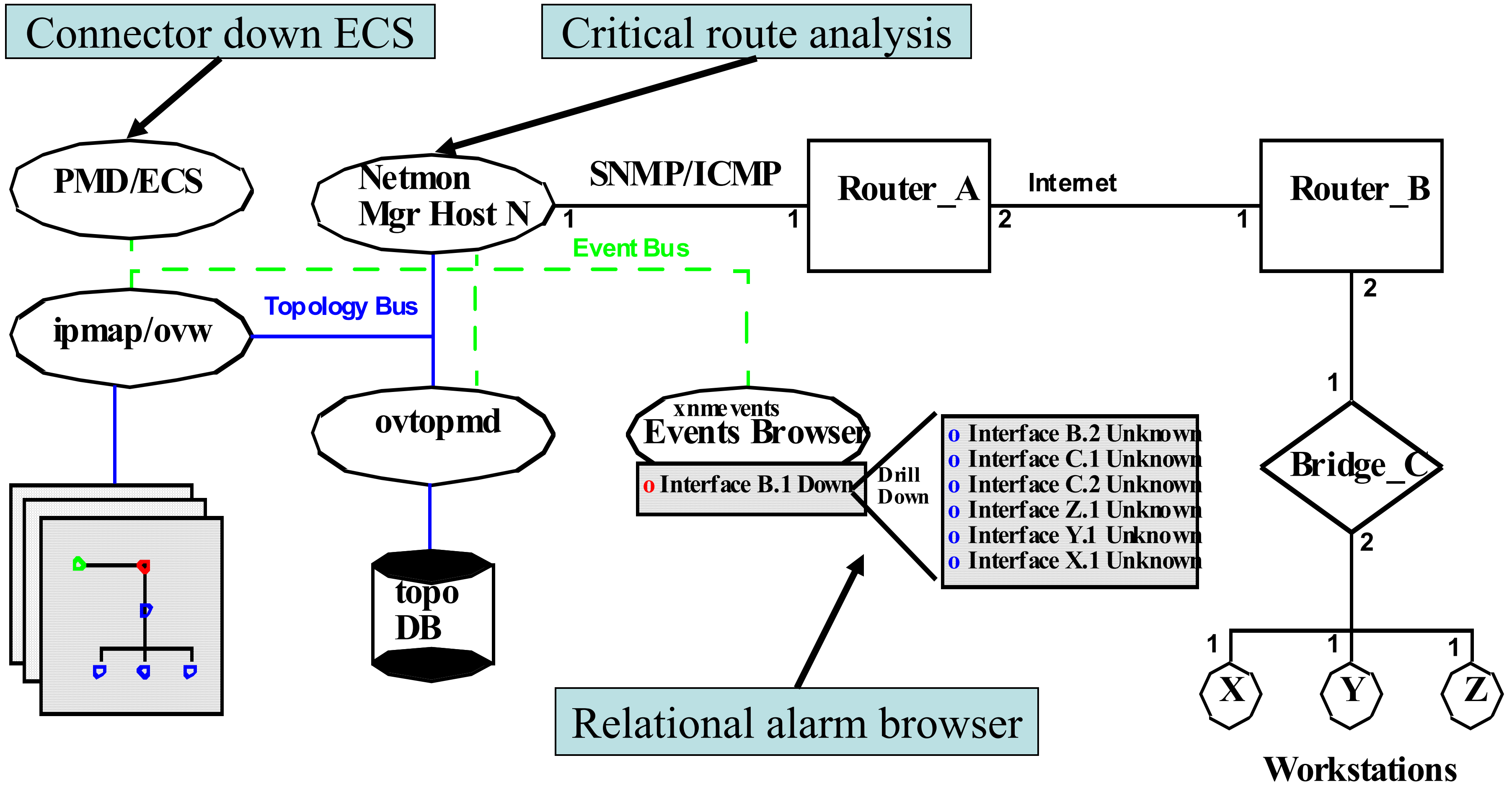
Summary of event correlations affecting status events

- NNM 6.0:
  - Connector Down relates primary to secondary failures
    - Node_down events show related IF_down events
  - Repeated Event applied to Node_up
  - Pair-wise applied to many status events
    - Alarms acknowledged if up condition occurs within 10 minutes
- NNM 6.31:
  - NodeIF supplements Connector Down
  - Pair-wise behavior updated & IntermittentStatus added
- NNM 6.4
  - De-duplication applied to status events
  - Status intervals, some ECS circuit parameters opened up

Cascade failure status event handling

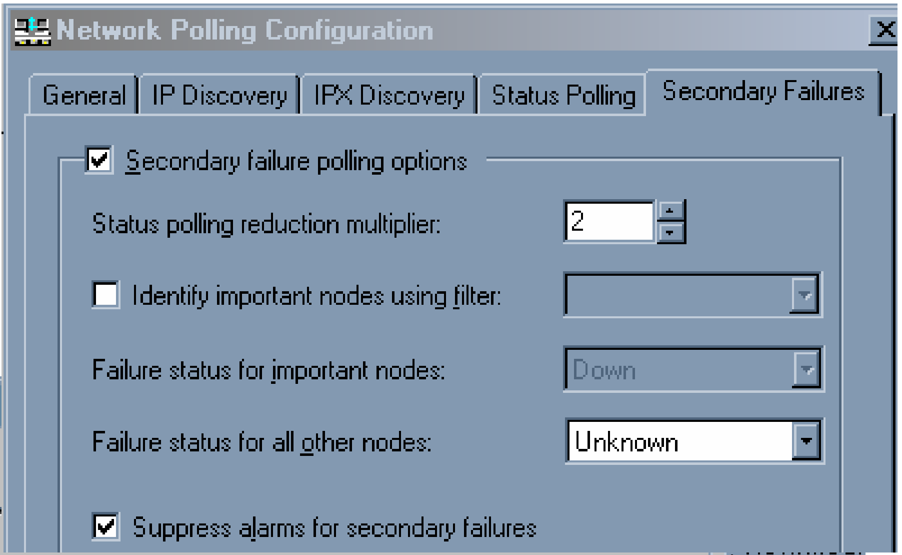
- Introduced NNM 6.0 through several major functionality adds:
  - Critical route analysis in *netmon*
    - *netmon* builds in-memory path to every interface
    - Distinguish primary and secondary failures
      - New varbind added to status events: ECS UUID of Primary
    - Reduce polls to secondary failures if not important nodes
      - Important Nodes filters defined using standard NNM FDL
  - ECS runtime engine with Connector Down correlation
    - Suppresses secondary failure events unless important
  - Relational event store (SOLID) - trapd.log deprecated
    - Show Correlated Events to view suppressed alarms

Cascade failure status event handling



Connector Down configuration entry points

- Network Polling Config:
- Event Correlation:


Network Polling Configuration

General | IP Discovery | IPX Discovery | Status Polling | Secondary Failures

☑ Secondary failure polling options

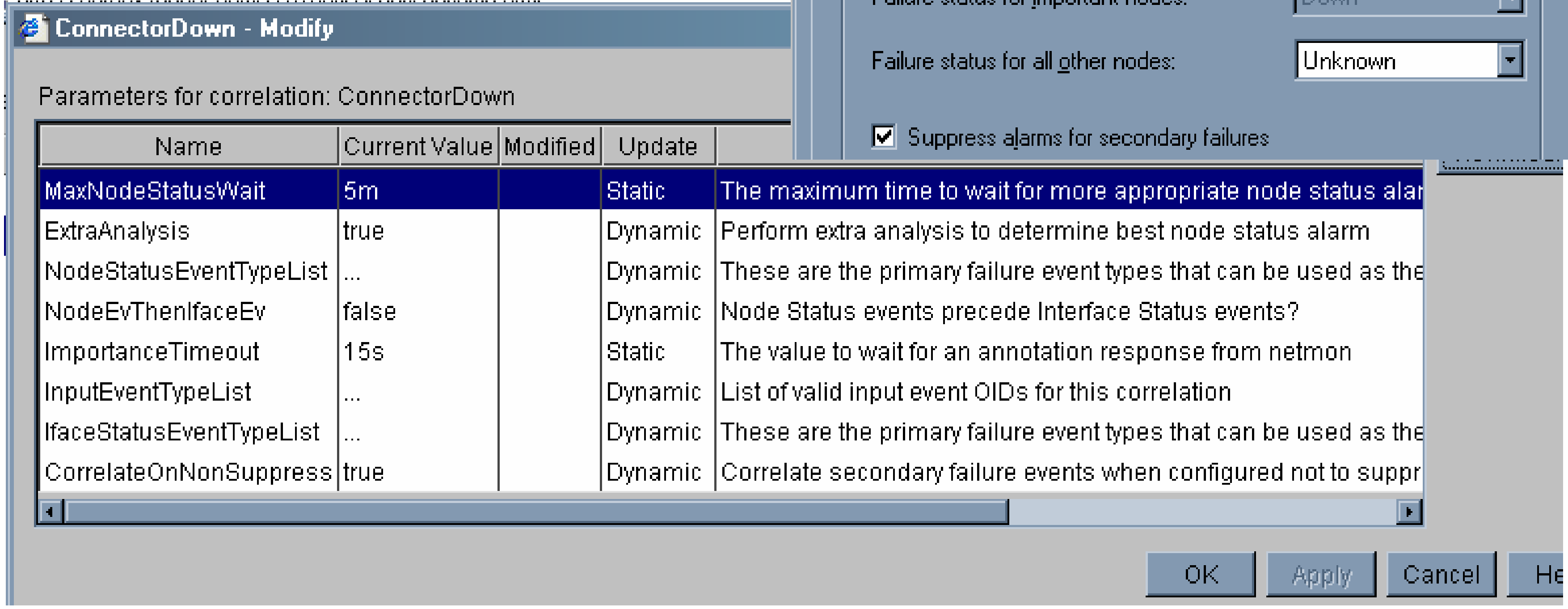
Status polling reduction multiplier: 2

☐ Identify important nodes using filter:

Failure status for important nodes: Down

Failure status for all other nodes: Unknown

☑ Suppress alarms for secondary failures


ConnectorDown - Modify

Parameters for correlation: ConnectorDown

Name	Current Value	Modified	Update	
MaxNodeStatusWait	5m		Static	The maximum time to wait for more appropriate node status alar
ExtraAnalysis	true		Dynamic	Perform extra analysis to determine best node status alarm
NodeStatusEventTypeList	...		Dynamic	These are the primary failure event types that can be used as the
NodeEvThenIfaceEv	false		Dynamic	Node Status events precede Interface Status events?
ImportanceTimeout	15s		Static	The value to wait for an annotation response from netmon
InputEventTypeList	...		Dynamic	List of valid input event OIDs for this correlation
IfaceStatusEventTypeList	...		Dynamic	These are the primary failure event types that can be used as the
CorrelateOnNonSuppress	true		Dynamic	Correlate secondary failure events when configured not to suppr

Pair-wise correlation

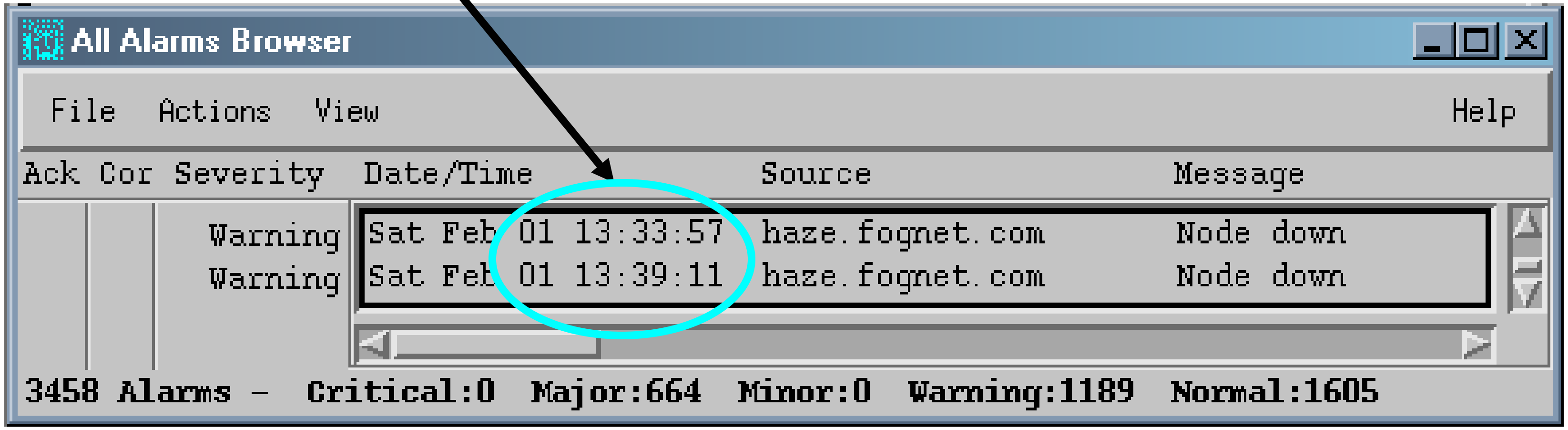
- Introduced in NNM V6.0
  - Applied to all status events logged (next slide)
  - Status alarms ***acknowledged*** if parent rec'd in PairedTimeWindow (10m)
  - Child events released immediately; Automatic actions launched
  - No reduction in the number of status events seen in alarm browser
- Behavior changed in NNM V6.31
  - Status alarms ***deleted*** if parent rec'd in PairedTimeWindow (10 minutes)
  - Child events held. Event only seen in alarm browser after 10 minute delay
  - Automated actions not launched prematurely, paired events never seen.
  - Details of parameters changed:
    - DeleteOrAcknowledge changed from AutoAcknowledge to Delete
    - ChildEventImmediateOutput=false
    - InhibitParentOfInhibitedChild=true

Status events affected by Pair-wise

- V6.0-V6.31:
  - Node up acknowledges Node: _Marginal, _Warning, _Major, _Down
  - Segment_Normal acknowledges Segment_Major, Segment_Critical
  - Network_Normal acks Network_Warning, Network_Critical
  - Station_Normal acks Station: _Marginal, _Warning, _Major, _Critical
  - Remote_Mgr_Up acknowledges Remote_Mgr_Down
- V6.4:
  - IF_Up deletes IF_Down
  - Node_Up deletes Node_Down, Node_Unknown
  - Segment_Normal deletes Segment_Major, Segment_Critical
  - Network_Normal deletes Network_Major, Network_Critical
  - Station_Normal deletes Station: _Marginal, _Major, _Critical
  - Remote_Mgr_Up deletes Remote_Mgr_Down

Pair-wise correlation example 1

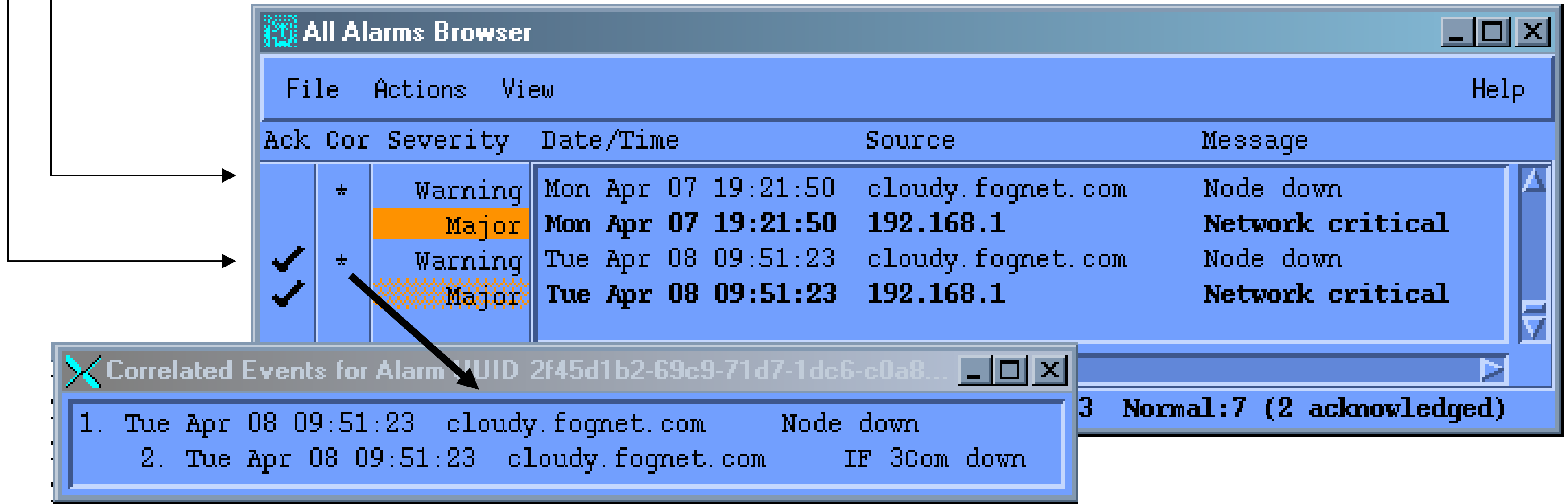
- All NNM versions prior to V6.0
  - Not in effect for any default status events
  - All status event released immediately and no stateful data held



```
Ack Cor Severity  Date/Time                 Source              Message
        Warning   Sat Feb 01 13:33:57       haze.fognet.com     Node down
        Warning   Sat Feb 01 13:39:11       haze.fognet.com     Node down
3458 Alarms - Critical:0 Major:664 Minor:0 Warning:1189 Normal:1605
```

Pair-wise correlation example 2

- NNM V6.0-V6.31
  - Node Down at 19:21; Node down event released immediately
    - PairedTimeWindow expires (10 minutes) – no action
  - Node Down at 19:51; If Up event received within 10 min
    - Pair-wise acknowledges message
    - Correlated events show ConnectorDown's correlated-events


All Alarms Browser
File Actions View Help
Correlated Events for Alarm UUID 2f45d1b2-69c9-71d7-1dc6-c0a8...

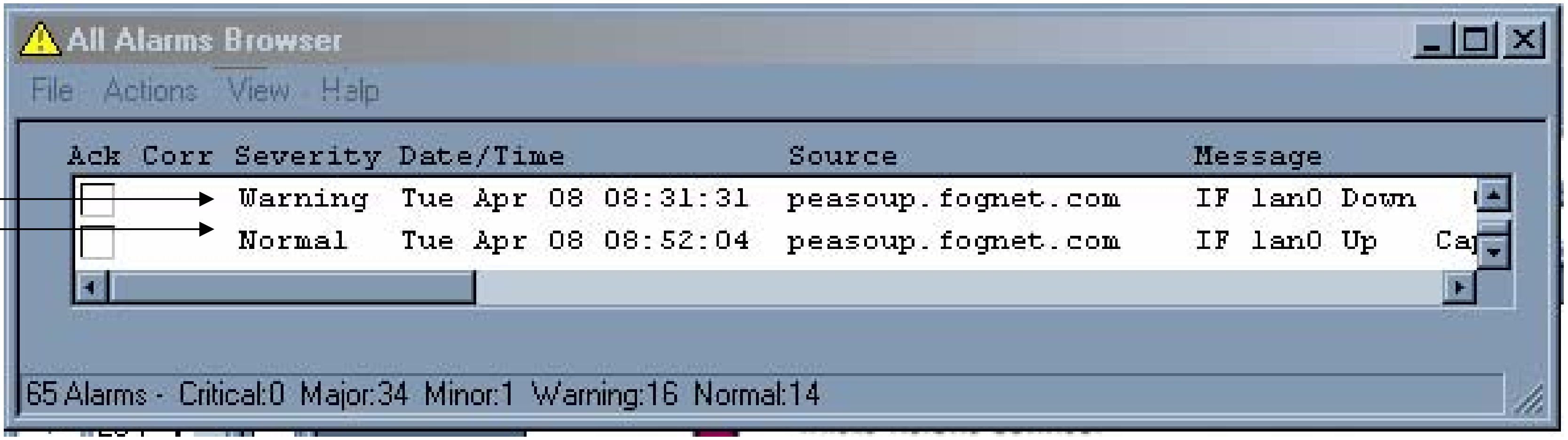
Ack	Cor	Severity	Date/Time	Source	Message
	*	Warning	Mon Apr 07 19:21:50	cloudy.fognet.com	Node down
		Major	Mon Apr 07 19:21:50	192.168.1	Network critical
✓	*	Warning	Tue Apr 08 09:51:23	cloudy.fognet.com	Node down
✓		Major	Tue Apr 08 09:51:23	192.168.1	Network critical

```
3  Normal:7 (2 acknowledged)
```

```
1.  Tue Apr 08 09:51:23  cloudy.fognet.com     Node down
    2.  Tue Apr 08 09:51:23  cloudy.fognet.com     IF 3Com down
```

Pair-wise correlation example 3

- V6.31+:
  - IF detected unreachable at 08:31
  - Event held until PairedTimeWindow expires (10 minutes by default)
  - IF_down event released to alarm browser at 08:41 with original timestamp
  - IF_up is a separate, un-correlated event

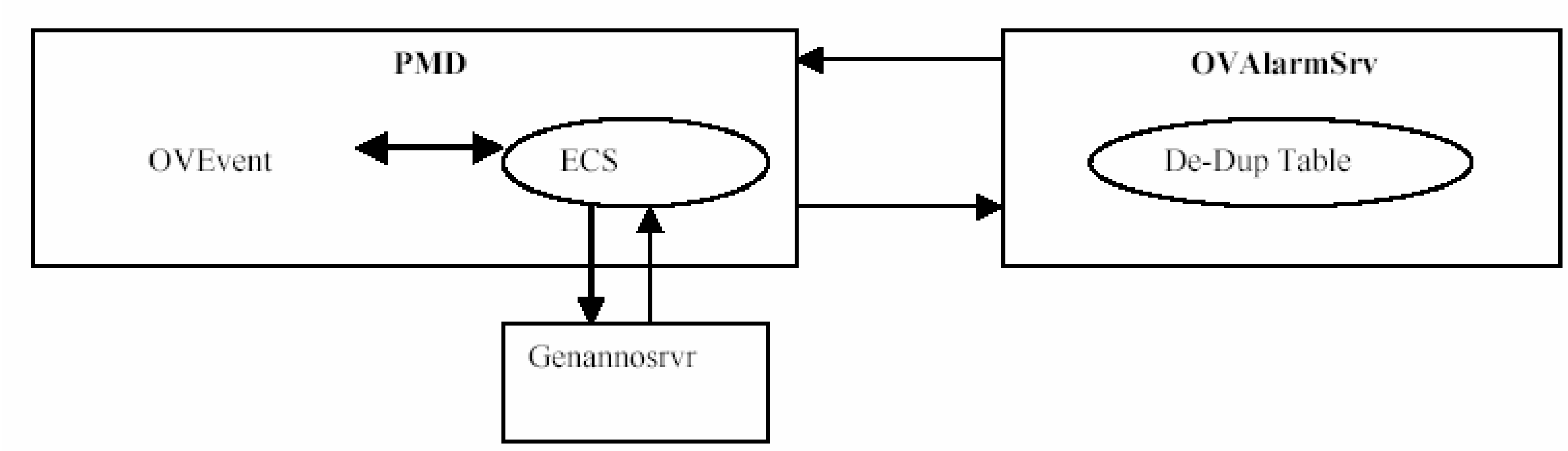


```
Ack Corr Severity Date/Time                Source                Message
         Warning  Tue Apr 08 08:31:31      peasoup.fognet.com    IF lan0 Down
         Normal   Tue Apr 08 08:52:04      peasoup.fognet.com    IF lan0 Up     Ca
65 Alarms - Critical:0 Major:34 Minor:1 Warning:16 Normal:14
```

- No alarms ever seen alarm browser if If_up received within 10 minutes

De-duplication (V6.4+)

- Dedup is a post-processing correlation,
  - Fed from *OVAlarmSRV* vs. *pmd* for other ECS circuits
  - Improves interaction with standard correlations effecting events
  - More info: NNM_Event_Reduction White Paper & dedup.conf man/ref
- Repeated event correlation becomes a “legacy” correlation
  - Repeated event default correlations affecting status:
    - OV_Node_Up in V6.0+ (RepeatedTimeWidow = 1h)

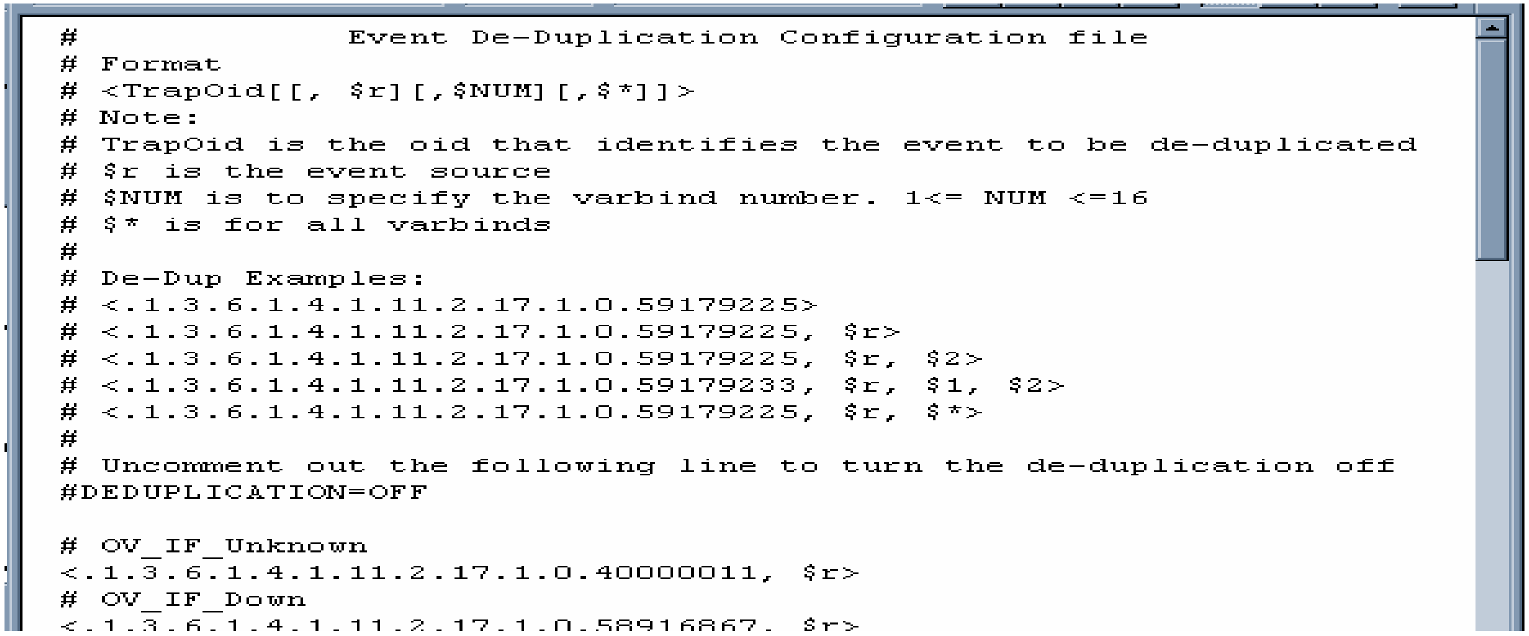


hp software forum 2003 WHERE VISIONS CONNECT

WHERE VISIONS CONNECT

De-duplication (V6.4+)

- Replaces existing identical event with latest; embedding previous
- $OV_CONF/dedup.conf is only configuration entry point
- Status events configured for de-dup by default:
  - OV_IF_Down, OV_IF_Unknown, OV_IF_Intermittent



```
#              Event De-Duplication Configuration file
# Format
# <TrapOid[[, $r][,$NUM][,$*]]>
# Note:
# TrapOid is the oid that identifies the event to be de-duplicated
# $r is the event source
# $NUM is to specify the varbind number. 1<= NUM <=16
# $* is for all varbinds
#
# De-Dup Examples:
# <.1.3.6.1.4.1.11.2.17.1.0.59179225>
# <.1.3.6.1.4.1.11.2.17.1.0.59179225, $r>
# <.1.3.6.1.4.1.11.2.17.1.0.59179225, $r, $2>
# <.1.3.6.1.4.1.11.2.17.1.0.59179233, $r, $1, $2>
# <.1.3.6.1.4.1.11.2.17.1.0.59179225, $r, $*>
#
# Uncomment out the following line to turn the de-duplication off
#DEDUPLICATION=OFF

# OV_IF_Unknown
<.1.3.6.1.4.1.11.2.17.1.0.40000011, $r>
# OV_IF_Down
<.1.3.6.1.4.1.11.2.17.1.0.58916867, $r>
```

NodeIf correlation (V6.31+)

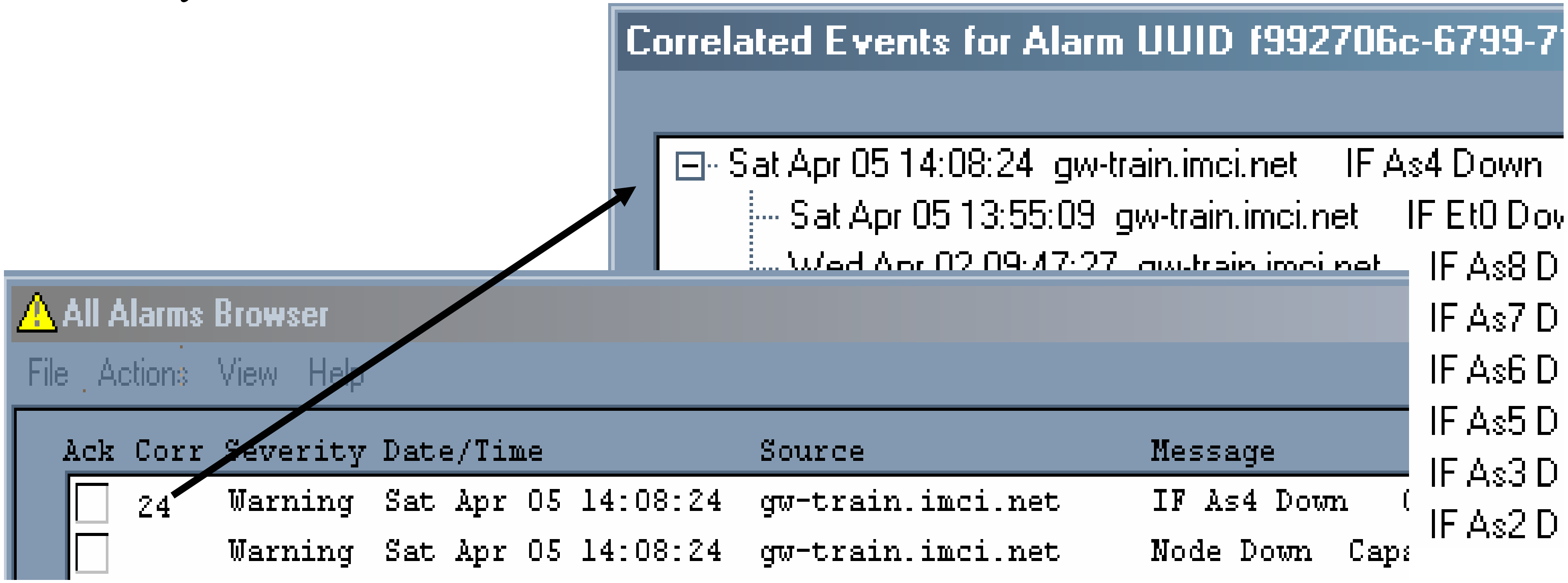
- Part of “node bias” paradigm shift Why? What was broken?
- For multiple interface devices:
  - Individual interface status “log-only” prior to V6.31
    - Connector device IF events not seen in alarm browser
      - Intermediate node status alarms OK, but not complete
    - Node up unpredictable, for example:
      - Node goes down, most interfaces come up – no node up event
      - Node up Repeated Event correlation a band-aid
- For single interface devices:
  - Interface status always directly maps to node status
  - Too many events if interface logging behaviors changed

NodeIf circuit characteristics (V6.31+)

- Supplements Connector Down correlation
  - Connector Down provides event suppression of secondary failures
  - NodeIf provides event suppression of connector nodes' interface events
  - Router/switch interface status events are correlated to the node event
- Uses data from new status event varbinds passed by *netmon*
  - Distinguishes simple devices (non-connector) from connector devices
  - Looks at # of If's and object capabilities (isIPRouter and isSwitch)
- Connector Device behavior:
  - Gather interface events; wait up to PairedTimeWindow (10 minutes);
  - Send resultant node status event unless node down or unknown is detected
- Simple Device behavior:
  - Send interface status events immediately, always suppress node status
- Coupled to Pair-wise correlation's PairedTimeWindow

NodeIf circuit event display

- One event in Alarm Browser
  - IF_down, or IF_Unknown
  - Related interface events embedded in Actions -> Correlated Events
  - of by double-click on number of events embedded



Correlated Events for Alarm UUID f992706c-6799-7

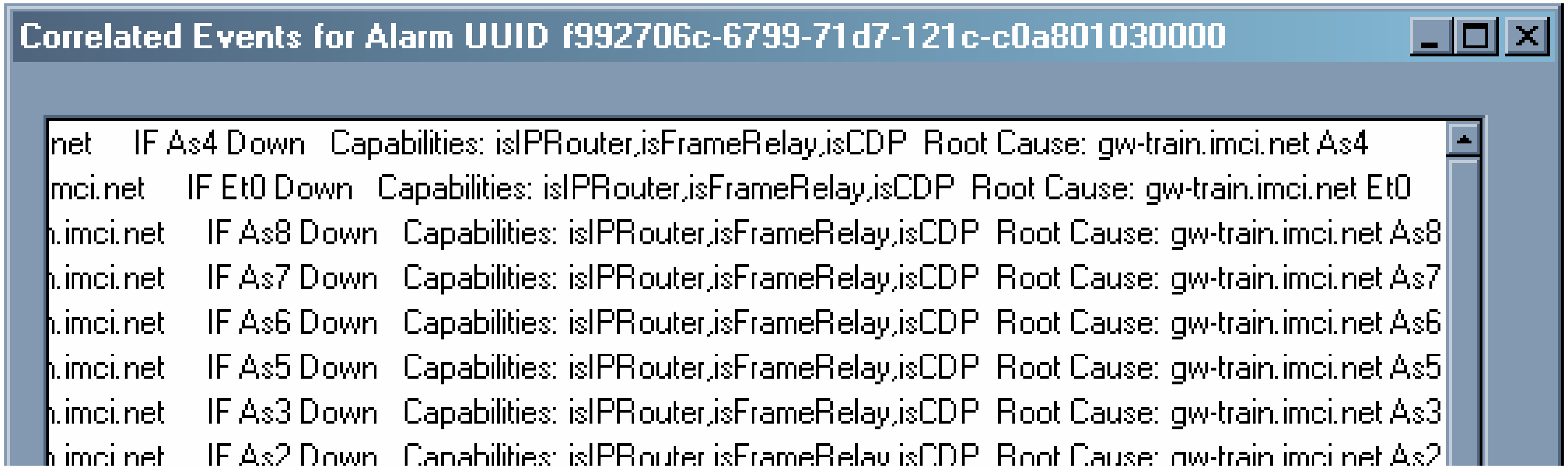
```
⊟ Sat Apr 05 14:08:24  gw-train.imci.net    IF As4 Down
    Sat Apr 05 13:55:09  gw-train.imci.net    IF Et0 Dov
    Wed Apr 02 09:47:27  gw-train.imci.net    IF As8 D
                                              IF As7 D
                                              IF As6 D
                                              IF As5 D
                                              IF As3 D
                                              IF As2 D
```

All Alarms Browser

```
Ack Corr Severity Date/Time            Source              Message
    24   Warning  Sat Apr 05 14:08:24  gw-train.imci.net   IF As4 Down    (
         Warning  Sat Apr 05 14:08:24  gw-train.imci.net   Node Down  Capa
```

Status event enhancements supporting NodeIf

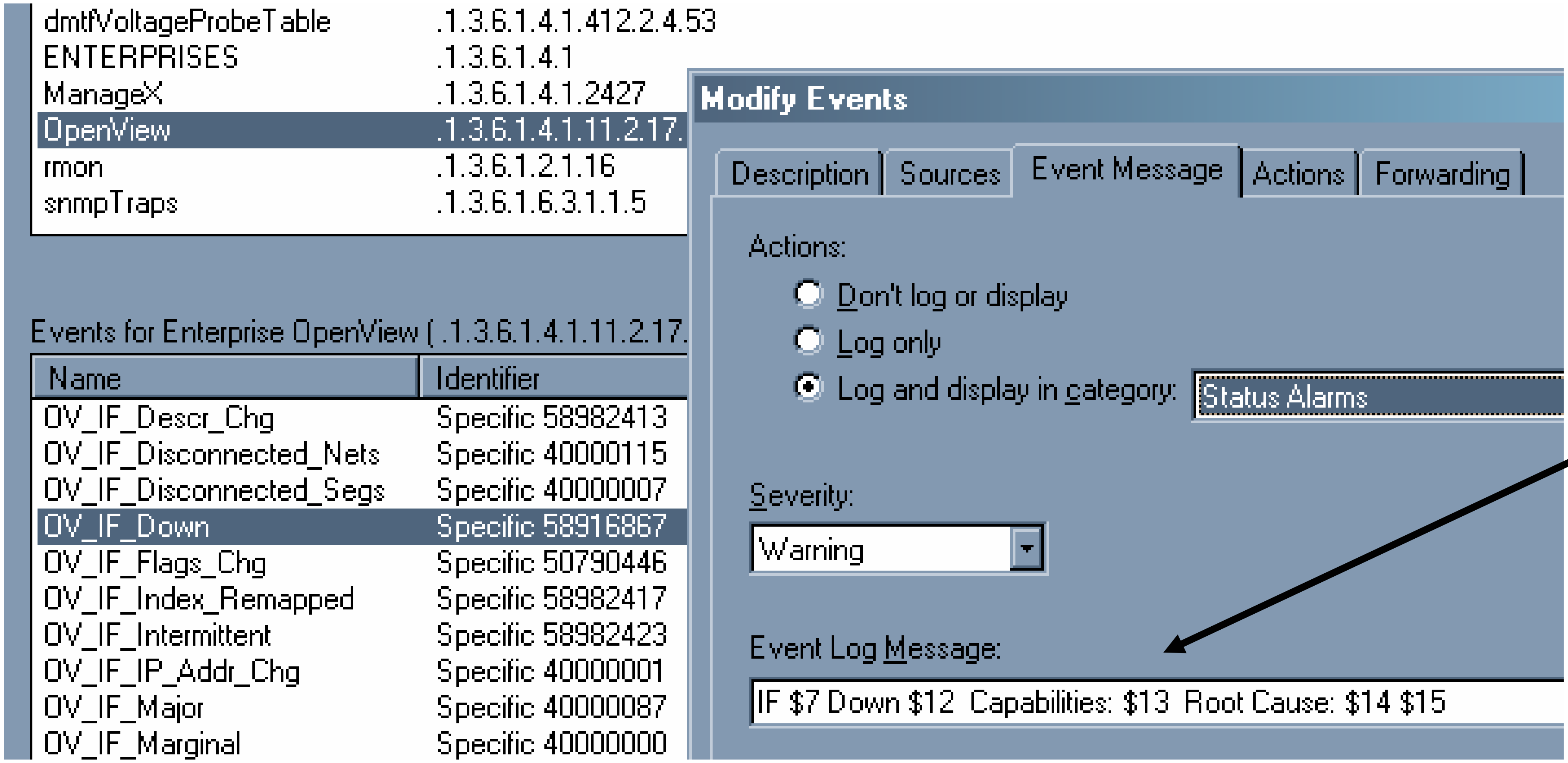
- First major change to default status event behaviors since V3.31
- New varbinds convey status of related primary failure entities
  - Derived from *netmon*/standard connector-down event correlation
  - Used in Alarm Message text to convey “root cause”
- New varbinds convey capabilities of related primary failure entities
  - Allows correlations to be tuned depending on device-type



Correlated Events for Alarm UUID f992706c-6799-71d7-121c-c0a801030000

```
net      IF As4 Down   Capabilities: isIPRouter,isFrameRelay,isCDP  Root Cause: gw-train.imci.net As4
mci.net    IF Et0 Down   Capabilities: isIPRouter,isFrameRelay,isCDP  Root Cause: gw-train.imci.net Et0
.imci.net    IF As8 Down   Capabilities: isIPRouter,isFrameRelay,isCDP  Root Cause: gw-train.imci.net As8
.imci.net    IF As7 Down   Capabilities: isIPRouter,isFrameRelay,isCDP  Root Cause: gw-train.imci.net As7
.imci.net    IF As6 Down   Capabilities: isIPRouter,isFrameRelay,isCDP  Root Cause: gw-train.imci.net As6
.imci.net    IF As5 Down   Capabilities: isIPRouter,isFrameRelay,isCDP  Root Cause: gw-train.imci.net As5
.imci.net    IF As3 Down   Capabilities: isIPRouter,isFrameRelay,isCDP  Root Cause: gw-train.imci.net As3
.imci.net    IF As2 Down   Capabilities: isIPRouter,isFrameRelay,isCDP  Root Cause: gw-train.imci.net As2
```

V6.31+ status event configuration



dmtfVoltageProbeTable	.1.3.6.1.4.1.412.2.4.53
ENTERPRISES	.1.3.6.1.4.1
ManageX	.1.3.6.1.4.1.2427
OpenView	.1.3.6.1.4.1.11.2.17.
rmon	.1.3.6.1.2.1.16
snmpTraps	.1.3.6.1.6.3.1.1.5

Events for Enterprise OpenView (.1.3.6.1.4.1.11.2.17.

Name	Identifier
OV_IF_Descr_Chg	Specific 58982413
OV_IF_Disconnected_Nets	Specific 40000115
OV_IF_Disconnected_Segs	Specific 40000007
OV_IF_Down	Specific 58916867
OV_IF_Flags_Chg	Specific 50790446
OV_IF_Index_Remapped	Specific 58982417
OV_IF_Intermittent	Specific 58982423
OV_IF_IP_Addr_Chg	Specific 40000001
OV_IF_Major	Specific 40000087
OV_IF_Marginal	Specific 40000000

Modify Events

Actions:
- Don't log or display
- Log only
- [✓ Log and display in category:] Status Alarms

Severity: Warning

Event Log Message:

```
IF $7 Down $12  Capabilities: $13  Root Cause: $14 $15
```

New varbinds in NNM 6.31+ status events

IF Status Varbind #	Node Status Varbind #	Description
$11		Number of bits in the interface subnet mask
$12		Interface ifAlias (hooray!)
$13	$8	Local list of capabilities
$14	$9	Name of primary failure host
$15	$10	Name of primary failure entity
$16	$11	OV OID of primary failure entity
$17	$12	Description of primary failure entity
$18	$13	Primary failure entity list of capabilities

NodeIf ECS circuit configuration

- PairedTimedWindow is only available parameter
- V6.31 “internal” correlations
  - Three circuits added that are not configurable from Event Correlation GUI
    - NodeIf, IntermittentStatus, Chassis
  - Configured through edit of .ds files in $OV_CONF/ecs/circuits/internal
  - “Managing” guide has procedure for reverting to previous version behavior
  - UNIX: NodeIf.ds is symbolic link to PairWise.ds, thus settable from GUI
- V6.4+ Correlation Composer
  - Launch from legacy Event Correlation GUI to configure NodeIf correlation
  - Allows “role-your-own” ECS circuits. (hooray!)
- NodeIf correlation structure:
  - Main correlation: OV_NodeIf_NodeDown – correlates IF to Node events
  - Two “helper” correlations:
    - OV_NodeIf_PrimaryIfUnknown – toss spurious unconnected IF events
    - OV_NodeIf_NodeNotConnector – toss Node events (simple devices)

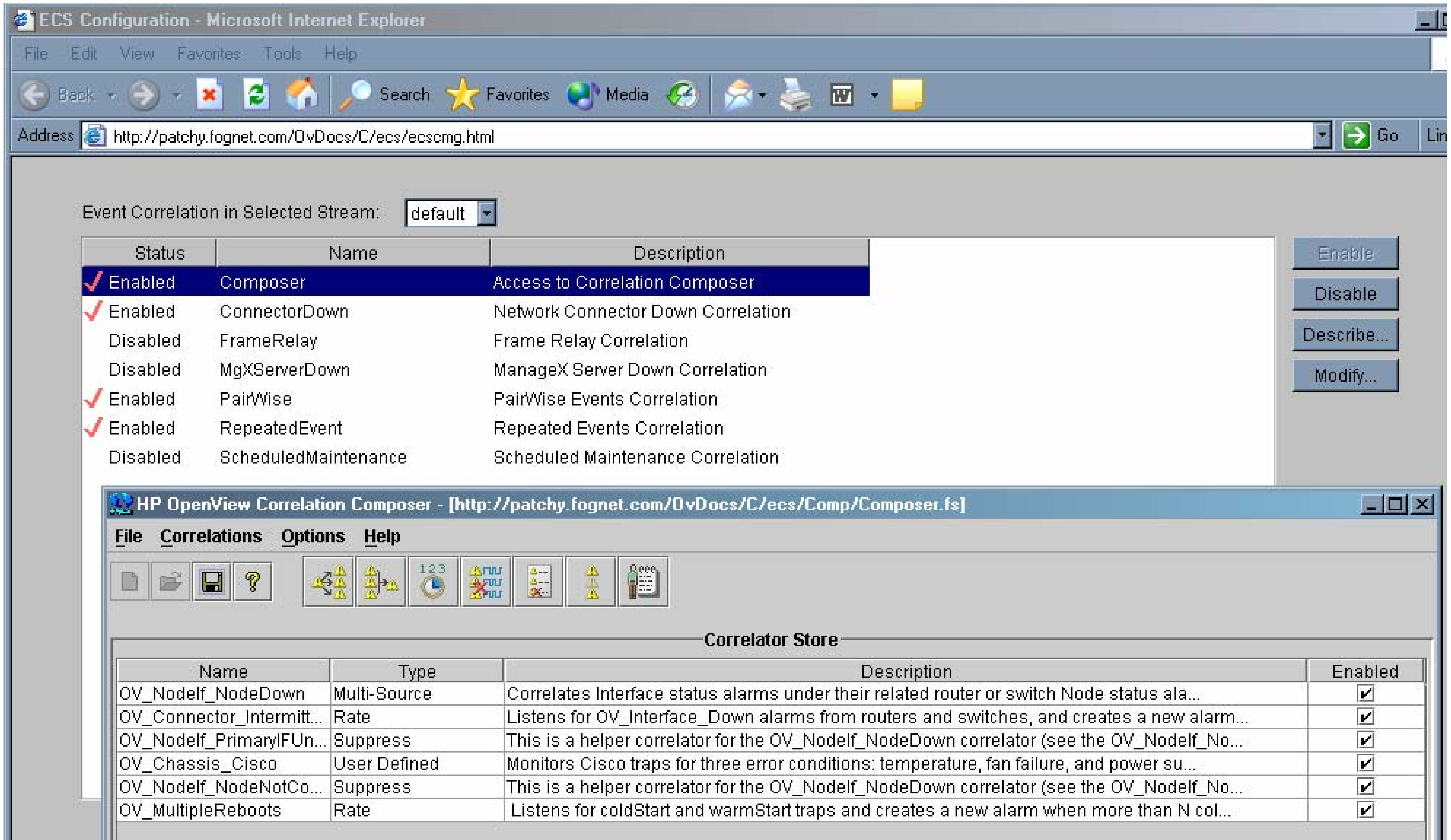
IntermittentStatus correlation

- New correlation in NNM V6.31+
  - Provides visibility to Pair-wise suppressions that are repeating
  - Applies only to connector device interfaces as determined by *netmon*
- New status event
  - OV_IF_Intermittent – OpenView enterprise 58982423
- Details
  - Generated if IF_down rec'd RATE_COUNT times over RATE_PERIOD
  - RATE_COUNT
    - Default is 4 in V6.31
    - Default is 5 in V6.4
  - RATE_PERIOD Default is 30 minutes
  - Loosely coupled to default status polling intervals
  - Loosely coupled to *netmon's* dynamic re-polls at 2 and 4 minutes

Correlation Composer (V6.4+)

- Premised on common logic sets for event-reducing correlations
  - A “Super circuit” providing sub-circuits:
    - Suppress, enhance, rate, repeated, transient, multiple source
  - Correlations defined under this model are called “instances”
- Not a replacement for default Event Correlation GUI
  - Note new “regular” correlation added in V6.4: FrameRelay
- More:
  - $OV_WWW/htdocs/C/manuals/COMPOSER.pdf
  - $OV_DOC/WhitePapers/Developing_NNM_Event_Reduction.pdf
- Event reduction tools in ascending order of complexity:
  - Event Customization, e.g. set event log-only or ignore
  - De-duplication
  - Composer
  - ECS Circuit customization

Event correlation configuration (V6.4+)


ECS Configuration - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://patchy.fognet.com/OvDocs/C/ecs/ecscmg.html Go
Enable
Disable
Describe...
Modify...
HP OpenView Correlation Composer - [http://patchy.fognet.com/OvDocs/C/ecs/Comp/Composer.fs]
File Correlations Options Help

Event Correlation in Selected Stream: default

Status	Name	Description
✓ Enabled	Composer	Access to Correlation Composer
✓ Enabled	ConnectorDown	Network Connector Down Correlation
Disabled	FrameRelay	Frame Relay Correlation
Disabled	MgXServerDown	ManageX Server Down Correlation
✓ Enabled	PairWise	PairWise Events Correlation
✓ Enabled	RepeatedEvent	Repeated Events Correlation
Disabled	ScheduledMaintenance	Scheduled Maintenance Correlation

Correlator Store

Name	Type	Description	Enabled
OV_NodeIf_NodeDown	Multi-Source	Correlates Interface status alarms under their related router or switch Node status ala...	✓
OV_Connector_Intermitt...	Rate	Listens for OV_Interface_Down alarms from routers and switches, and creates a new alarm...	✓
OV_NodeIf_PrimaryIFUn...	Suppress	This is a helper correlator for the OV_NodeIf_NodeDown correlator (see the OV_NodeIf_No...	✓
OV_Chassis_Cisco	User Defined	Monitors Cisco traps for three error conditions: temperature, fan failure, and power su...	✓
OV_NodeIf_NodeNotCo...	Suppress	This is a helper correlator for the OV_NodeIf_NodeDown correlator (see the OV_NodeIf_No...	✓
OV_MultipleReboots	Rate	Listens for coldStart and warmStart traps and creates a new alarm when more than N col...	✓

Scenario: Simple device unreachable

- Computer with one interface goes down (NNM V6.31+)
    - Node_down event will ***never*** be generated, suppressed by NodeIf
    - IF_down event held by Pair-wise 10 minutes, ***then*** sent to alarm browser
        - Automatic actions launched only if If_up event not received in 10 min
        - No alarm sent at all if If_up event not received in 10 minutes
    - Primary interface down event embedded by ConnectorDown
    - Any existing If_down events in Alarm Browser embedded by dedup (V6.4)

- Computer with one interface goes down (NNM V6.0 up to V6.31)
    - ***Only*** Node_down event sent since all IF events log-only
    - Node_down sent to alarm browser immediately, automatic actions launched
    - Node down acknowledged if Node_up rec'd within 10 min by pair-wise

Scenario: Connector device interface outages

- One or more interfaces on a switch go down (V6.31+)
  - IF_down event held by NodeIF for PairedTimeWindow (10 minutes)
  - IF events suppressed/embedded if Node_down or Node_Unknown detected
  - Additional IF_down events for switch suppressed and embedded
  - Pair-wise in effect: if IF comes up within 10 mins, IF_down/up pair discarded
  - IntermittentStatus event sent if IF "flaps" 4 or 5 times in 30 minutes
  - Primary failure event embedded by ConnectorDown
  - Any existing If_down events in Alarm Browser embedded by dedup (V6.4)

- One or more interfaces on a switch go down (All previous versions)
  - Node status event ***may*** be generated if status propagation rules satisfied
  - Node_down or Node_unknown sent when last reachable interface goes down
  - Node_up sent when all interfaces reachable, Repeated Event in effect (V6.0+)
  - Pair-wise in effect: Node up within 10 mins, Node status event acknowledged

Summary of influences on *netmon* status polling interval (defaults)

- Global status polling settings (*xnmsnmpconf)*
  - 15 Minutes, 0.8 sec timeout, 2 retries
- Secondary failure reduction multiplier (NNM V6.0+)
  - Reduce polls to 2ndaries at scheduled status polling interval times two
- Object-based status (V6.2+)
  - Status polling intervals based on any NNM filter, e.g. Routers, Bridges
  - Status polling can optionally specified for primary interfaces
- Priority status polling (V6.31+)
  - All interfaces on connector devices polled immediately if one IF down
- Failure mode polling (V6.31+)
  - Any down interface re-polled at 2 minutes and 4 minutes
- Intermittent status polling (V6.31+)
  - Any newly up interface re-polled at 2 minutes

Summary of node and interface status events

- Default logging status and effect of Connector Down & NodeIf

		6.31-	6.31+
OV_Node_Up:	All interfaces up	**LO**	**LO**
OV_Node_Warning:	One interface down; others up or unknown	**L**	**LO**
OV_Node_Marginal:	>One interface down; others up or unknown	**L**	**LO**
OV_Node_Major:	One interface up	**L**	**LO**
OV_Node_Down:	All interfaces down or unknown.	**L***	**C***
OV_Node_Unknown:	All interfaces on the node are unknown	**L***	**C***
OV_IF_Down:	Interface Down	**LO**	**C***
OV_IF_Up:	Interface Up	**LO**	**LO**
OV_IF_Unknown:	Interface Unknown	**LO**	**C***

L – Logged   LO – Log-Only   C – Correlated by NodeIf   * ConnectorDown

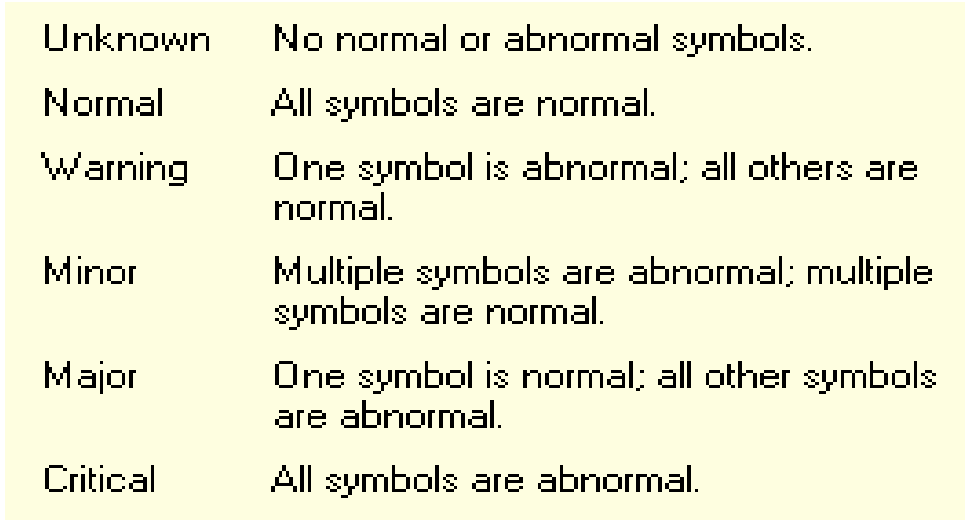
Summary of topology status

	6.31-	6.31+
OV_Segment_: Marginal, Normal, Warning, Unknown	**LO**	**LO**
OV_Segment_Major: One contained node normal	**L**	**LO**
OV_Segment_Critical: All contained nodes down or unknown	**L**	**LO**
OV_Network_: Marginal, Normal, Warning, Unknown	**LO**	**LO**
OV_Network_Major: All connectors & segments down or unknown	**L**	**LO**
OV_Network_Critical: One connector or segment down or unknown	**L**	**LO**

Other status events affecting topology

- Connection (all **LO**),
- Station_*
- Remote_Mgr_*
- IPV6_* (V6.4+)
- HSRP_* (V6.4+)

Default map status propagation:



Unknown	No normal or abnormal symbols.
Normal	All symbols are normal.
Warning	One symbol is abnormal; all others are normal.
Minor	Multiple symbols are abnormal; multiple symbols are normal.
Major	One symbol is normal; all other symbols are abnormal.
Critical	All symbols are abnormal.

Summary of status configuration entry points

- General Status Polling configuration
  - SNMP Configuration (*xnmsnmpconf*): Polling intervals
  - Network Polling Configuration GUI (*xnmpolling*): Polling behavior
  - *netmon.statusIntervals*: Polling by Filter Object
  - *netmon* switches (e.g. -k snmpTimeoutInmplies=status)
- Events
  - Event Configuration GUI (*xnmtrap*): Configure status alarms
  - *netmon.snmpStatus, netmon.statusMapping:* Configure SNMP Status
- ECS
  - Options -> Event Configuration -> Edit -> Event Correlation
- Maps
  - Map topology status propagation rules (Map -> Properties)
  - Symbol properties status source (object, symbol, compound)

Future enhancements affecting status polling that I've heard tell of

- NNM 6.5 (2004)
  - Rules-based polling
  - Additional event-map view integrations for drill-down

That's all folks…
Thank you!