

#SHAREorg

**z/OS Parallel Sysplex® Update
and New XCF Client/Server APIs**

Mark A Brooks
IBM
mabrook@us.ibm.com

11:00 a.m. Tuesday March 13, 2012
Session 10649 (long)

In this session, the speaker will provide updates on Parallel Sysplex, including the latest hardware and coupling link technology, Coupling Facility Control Code (CFCC), and recent z/OS enhancements. A preview of future z/OS items may be included as well. In the past we have heard on such topics as the physical processors, software levels, links to external coupling facilities, upgrade paths, and features in the CFCC code. Plus how Server Time Protocol (STP) fits in the picture. On the software side, what Program Products participate in a Sysplex and how that may increase application availability for you.

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

IBM®	MQSeries®	S/390®	z9®
ibm.com®	MVS™	Service Request Manager®	z10™
CICS®	OS/390®	Sysplex Timer®	z/Architecture®
CICSPlex®	Parallel Sysplex®	System z®	zEnterprise™
DB2®	Processor Resource/Systems Manager™	System z9®	z/OS®
eServer™	PR/SM™	System z10®	z/VM®
ESCON®	RACF®	System/390®	z/VSE®
FICON®	Redbooks®	Tivoli®	zSeries®
HyperSwap®	Resource Measurement Facility™	VTAM®	
IMS™	RETAIN®	WebSphere®	
IMS/ESA®	GDPS®		
	Geographically Dispersed Parallel Sysplex™		


The following are trademarks or registered trademarks of other companies.

IBM, z/OS, Predictive Failure Analysis, DB2, Parallel Sysplex, Tivoli, RACF, System z, WebSphere, Language Environment, zSeries, CICS, System x, AIX, BladeCenter and PartnerWorld are registered trademarks of IBM Corporation in the United States, other countries, or both.
 DFSMSshm, z9, DFSMSmm, DFSMSdtp, DFSMSdss, DFSMS, DFS, DFSORT, IMS, and RMF are trademarks of IBM Corporation in the United States, other countries, or both.
 Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
 Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
 InfiniBand is a trademark and service mark of the InfiniBand Trade Association.
 UNIX is a registered trademark of The Open Group in the United States and other countries.
 Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:
 Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
 IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.
 All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
 This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
 All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
 Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
 Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

2



For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml

Statements of Direction



- **z196 announcement, July 22, 2010**
- **z114 announcement, July 12, 2011**
- **The z196 and z114 will be the last System z servers to:**
 - **Offer ordering of ESCON channels**
 - **Offer ordering of ISC-3**
 - **Support dial-up modem**
- **Implications**
 - If using CTC devices for XCF signalling paths, need to migrate to FICON from ESCON
 - Migrate from ISC-3 coupling links to infiniband
 - Migrate to alternatives for dial-up time services

3



The IBM zEnterprise 196 and the IBM zEnterprise 114 are the last System z servers to support ESCON channels. IBM plans to not offer ESCON channels as an orderable feature on System z servers that follow the z196 (machine type 2817) and z114 (machine type 2818). In addition, ESCON channels cannot be carried forward on an upgrade to such followon servers. This plan applies to channel path identifier (CHPID) types CNC, CTC, CVC, and CBY and to feature #2323 and #2324.

System z customers should continue to eliminate ESCON channels from the mainframe wherever possible. Alternate solutions are available for connectivity to ESCON devices. IBM Global Technology Services offers an ESCON to FICON Migration solution, Offering ID #6948-97D, to help facilitate migration from ESCON to FICON. This offering is designed to help you to simplify and manage a single physical and operational environment - FICON channels on the mainframe with continued connectivity to ESCON devices.

The IBM zEnterprise 196 and the zEnterprise z114 will be the last servers to offer ordering of ISC-3. Enterprises should begin migrating from ISC-3 features (#0217, #0218, #0219) to 12x InfiniBand (#0163 - HCA2-O or #0171 - HCA3-O fanout) or 1x InfiniBand (#0168 - HCA2-O LR or #0170 - HCA3-O LR fanout) coupling links.

The z114 and z196 are planned to be the last servers to support dial-up modems for use with the Remote Support Facility (RSF), and the External Time Source (ETS) option of Server Time Protocol (STP). The currently available Network Time Protocol (NTP) server option for ETS as well as Internet time services available using broadband connections can be used to provide the same degree of accuracy as dial-up time services. Enterprises should begin migrating from dial-up modems to Broadband for RSF connections.

Note: *IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.*

Agenda



- Hardware Updates
 - CFCC Level 17
 - InfiniBand (IFB) coupling links
 - Server Time Protocol (STP)
- Software Updates
 - z/OS V1R13 (new XCF Client/Server APIs)
 - z/OS V1R12
 - z/OS V1R11
- Summary

4



Coupling Facility Control Code (CFCC) Level 17 is required when a Coupling Facility (CF) is instantiated on an IBM zEnterprise™ 196 (z196) or an IBM zEnterprise™ 114 (z114).

CFCC Level 16 is required for System z10.

CFCC Level 17 – Constraint Relief



- Up to 2047 structures per CF image (prior limit 1023)
 - Many data sharing groups, SAP, CICS, merging plexes
 - New version of CFRM CDS needed to define more than 1024 structures

APAR OA38840
- Supports up to 255 connectors for all structure types
 - Cache structures already support 255 connectors
 - z/OS imposes smaller limits for lock structures (247) and serialized list structures (127)
 - Will require exploiter changes as well (none yet!)
- Prerequisites
 - z/OS V1.10 or later with PTF for OA32807
 - z/VM V5.4 for guest virtual coupling

5



Coupling Facility Control Code (CFCC) Level 17 supports **up to 2047 structures per Coupling Facility (CF) image** (vs. prior limit of 1023 for CFCC Level 16). The new limit helps when a large number of structures must be defined. For example, installations with large numbers of data sharing groups (e.g., SAP configurations), or large numbers of CICS regions using CF log streams, or who are merging sysplexes (a single CF may need to hold all structures when a failure occurs). This function requires z/OS 1.10 or later with the PTFs for APAR OA32807. A new version of the z/OS Coupling Facility Resource Manager (CFRM) Couple Data Set (CDS) is required in order to define more than 1024 CF structures to z/OS. z/OS allows up to 2048 structures to be defined, but at most 2047 could be allocated due to the smaller CF limit. APAR **OA38840** fixes a situation where a system wait-states due to fragmentation of real storage below the 16M line when IPLing with a CFRM CDS formatted to support 2048 structures and 255 connectors per structure. The error scenario applies only if the IPLing system is configured to use CF structures for XCF signalling. It could occur for smaller limits as well. Occurrence of the problem depends on the timing of when and the degree to which storage becomes fragmented.

CFCC Level 17 supports **255 connectors to all structure types**. Both z/OS and CFCC already support 255 connectors to cache structures. With this new support XES also supports up to 247 connectors to a lock structure, 127 connectors to a serialized list structure, and 255 connectors to an unserialized list structure. This support requires z/OS 1.10 or later with the PTFs for APAR OA32807.

In the early days of sysplex, the norm was 1 z/OS = 1 DB2. With a max of 32 systems in a sysplex, it was natural to assume a maximum of 32 DB2s in a data sharing group. For various reasons, we now see cases where many DB2 subsystems from the same data sharing group reside in the same z/OS system. To support this move to larger data sharing groups, CFCC Level 17 increases the number of connectors to a list or lock structure from 32 to up to 255. The z/OS exploitation of these structures imposes smaller limits than what the CF supports. For lock structures, 8 bits of the 32 byte lock table entry is consumed for “global ownership”, leaving only 247 bits to represent shared interest. For a serialized list, one bit of the one byte lock table entry is used for lock management, so x'7F' = 127 is the largest number of connectors that can be represented in the remaining bits of the byte.

CFCC Level 17 - Serviceability



- CF Diagnostics

- Non-disruptive dumping
- Coordinated dump capture
 - Gathers z/OS, CF, and link diagnostics at same time
 - Use DUPLEXCFDIAG to enable

Also available for z10
CFCC Level 16 (need MCLs)
z/OS APAR OA33723

- Prerequisites

- z/OS V1.12
- z/OS 1.10 or 1.11 with PTFs for OA31387

6



Non-disruptive CF dumping

With z/OS V1.12, or with the PTFs for OA31387 installed on z/OS V1.10 or z/OS V1.11, in conjunction with z196 servers and Coupling Facility control code (CFCC) Level 17, the system can capture serialized Coupling Facility (CF) data nondisruptively. This new function is intended to help improve Parallel Sysplex availability when it is necessary to capture CF data. The CF uses a pre-staged dump capture area to avoid collateral effects observable by z/OS, such as message time-outs (observed as interface control checks) or loss of connectivity.

Prior to this support, there were two ways to capture diagnostic information from the CF. The most useful way required a disruptive CF hard dump, requiring a reboot of the CF and lossconn recovery (rebuilds, etc.) from z/OS. Understandably, customers were rarely willing to install diagnostic CFCC code loads to produce such a dump. The other method was a CFCC soft dump, which produces a “blurry”, unserialized picture because CF activity is not quiesced while the dump is captured. In many cases, the soft dumps were of little use for problem diagnosis. The new support enables the capture of useful diagnostic data without impact to the sysplex.

Coordinated Dump Capture

This support introduces the ability to initiate the collection of end-to-end data for duplexing-related message time-out conditions. When running on an IBM zEnterprise 196 (z196) machine, and when the two CFs involved in the duplexed request are at or above CFLEVEL 17, z/OS will be able to trigger collection of documentation from both CFs, all z/OS images hosted on z196 machines and connected to those CFs, and the links connecting the z/OS images to the CFs and the CFs to each other. By default, this data collection is disabled. When desired, installations can initiate this data collection using the SETXCF FUNCTIONS,ENABLE=DUPLEXCFDIAG command or the FUNCTIONS ENABLE(DUPLEXCFDIAG) COUPLExx parmlib member statement.

When the DUPLEXCFDIAG function is enabled, software and hardware problem determination data will be collected when a duplexed coupling facility (CF) request is delayed in completion. The intent of this support is to collect coherent data from z/OS, coupling facilities, and message paths when it appears that a CF may be about to break duplexing for the structure with which the delayed command is associated. The function is available only when the structure is duplexed between two CFs at CFLEVEL 17 or above, and when the link firmware supports the data collection function.

CFCC Level 17 - Migration



- In general, get to most current LIC levels
- Use CF Sizer website to check/update structure sizes:
 - CF structure sizes may increase when migrating to CFCC Level 17 from earlier levels due to additional CFCC controls
 - IBM's testers saw 0-4% growth from CFLEVEL=16
 - Improperly sized structures can lead to outages !
- Minimum CFCC image size is now 512MB

www.ibm.com/systems/support/z/cfsizer/

7



CFCC Level 17 is expected to have marginal impact to structure sizes. Representative, properly sized structures used by IBM testers grew between 0 and 4% when allocated in a coupling facility running CFLEVEL 17 vs a coupling facility running CFLEVEL 16. These results may not apply to your environment. Thus IBM suggests using the CF Sizer to size structures whenever a new CF level is installed. The CF Sizer can be found at this web site:

www.ibm.com/systems/support/z/cfsizer/index.html

The web-based CFSIZER tool is designed to help you perform coupling facility structure sizing calculations. Concurrently with z/OS V1.12 availability, CFSIZER provides:

- More accurate size calculations for the IMS Operations Management (OM) Audit, Resource, and Overflow Sequential Access Method (OSAM) structures
- Calculated sizes for IBM Session Manager (ISM) Sysplex User structures
- Improved sizes for XCF signaling structures
- Calculated sizes for InfoSphere™ Classic control and filter structures
- Improved sizes for DB2 SCA structures
- Various usability improvements to the CFSIZER Web pages, including consolidated structure input pages to the OEM cache structure page.

The storage requirement for the CFCC image size is now 512MB (previously 128MB). That is the amount of storage needed just to boot the CF Control Code. Additional storage is needed to allocate the actual structures. You need to allow for growth of the CFCC image size as well as the growth in the size of the structures when configuring the LPAR for your CF.

Agenda



- Hardware Updates
 - CFCC Level 17
 - InfiniBand (IFB) coupling links
 - Server Time Protocol (STP)
- Software Updates
 - z/OS V1R13
 - z/OS V1R12
 - z/OS V1R11
- Summary

8



Glossary for System z Coupling		
Acronym	Full name	Comments
AID	Adapter identification	HCA fanout has AID instead of a PCHID
CIB	Coupling using InfiniBand	CHPID type z196, z10, System z9
HCA	Host Channel Adapter	Path for communication
MBA	Memory Bus Adapter	Path for communication
PSIFB	Parallel Sysplex using InfiniBand	InfiniBand Coupling Links
12x IFB	12x InfiniBand	12 lanes of fiber in each direction
1x IFB	1x InfiniBand	Long Reach one pair of fiber
12x IFB3	12x InfiniBand3	Improved service times versus 12x IFB on HCA3-O

9

SHARE in Atlanta 2012

This chart describes the basic terminology used when discussing coupling links. If you are unfamiliar with these terms, the Redbook "IBM System z Connectivity Handbook" (SG24-5444) is a good place to start. See Chapter 10 which discusses coupling technology.

InfiniBand (IFB) coupling links are high-speed links on z196, z114, and System z10. The IFB coupling links originate from four types of fanouts:

HCA3-O (FC 0171) for 12x IFB links on zEnterprise CPCs

HCA3-O LR (FC 0170) for 1x IFB links on zEnterprise CPCs

HCA2-O (FC 0163) for 12x IFB links on zEnterprise CPCs and System z10 servers

HCA2-O LR (FC 0168) for 1x IFB links on zEnterprise CPCs1 and System z10 servers.

Each fanout used for coupling links has an assigned adapter ID number (AID) that must be used for definitions in IOCDS to have a relation between the physical fanout location and the CHPID number.

12x IFB coupling links

The HCA2-O and HCA3-O fanout support IFB coupling links that operate at 6 Gbps (12x IFB). IFB coupling links use a fiber optic cable that is connected to a HCA2-O or HCA3-O fanout. The maximum distance for 12x IFB link is 150 meters. There are two protocols supported by the HCA3-O for 12x IFB feature. **12x IFB protocol:** If more than four CHPIDs are defined per HCA3-O port, or HCA3-O features are communicating with HCA2-O features on zEnterprise or System z10 servers, links will run with the 12x IFB protocol. **12x IFB3 protocol:** When HCA3-O fanouts are communicating with HCA3-O fanouts and have been defined with four or fewer CHPIDs per port, the 12x IFB3 protocol is utilized. The protocol has been designed to deliver improved link services times. In particular, synchronous link service times are designed to be 40% faster than when using the 12x IFB protocol.

1x IFB coupling links The HCA2-O LR and HCA3-O LR fanout support 1x IFB coupling links that operate at up to 5.0 Gbps. 1x IFB coupling links use a fiber optic cable that is connected to a HCA2-O LR or HCA3-O LR fanout. The maximum unrepeated distance for a 1x IFB link is 10 km. When using repeaters, the maximum distance is up to 100 km. 1x IFB supports either 7 or 32 subchannels.

Fanout adapter ID Unlike channels installed in an I/O cage, I/O drawer, or PCIe I/O drawer, which are identified by a PCHID number related to their physical location, coupling link fanouts and ports are identified by an adapter ID (AID). The adapter ID value depends on its physical location. The AID must be used to assign a CHPID to the fanout in the IOCDS definition. The CHPID assignment is done by associating the CHPID to an AID and port.

Glossary for System z Coupling ...



Type	System z10	System z196/z114
HCA1-O fanout	NA	NA
HCA2-O fanout	Optical - Coupling 12x InfiniBand	Optical - Coupling 12x InfiniBand
HCA2-O LR fanout	Optical - Coupling 1x InfiniBand	Optical - Coupling 1x InfiniBand
HCA3-O fanout	NA	Optical - Coupling 12x InfiniBand
HCA3-O LR fanout	NA	Optical - Coupling 1x InfiniBand
MBA fanout	Copper - Coupling (ICB-4)	N/A

10




This chart describes the basic terminology used when discussing coupling links. If you are unfamiliar with these terms, the Redbook “IBM System z Connectivity Handbook” (SG24-5444) is a good place to start. See Chapter 10 which discusses coupling technology.

(more terms on previous slide)


HCA n -O = Host Channel Adapter n 'th generation – Optical

Your coupling links (cables) plug into these adapters.

Coupling link choices



Short Distance	<ul style="list-style-type: none"> • IC (Internal Coupling Channel) <ul style="list-style-type: none"> • Microcode - no external connection • Only between partitions on same processor • ICB-3 and ICB-4 (Integrated Cluster Bus) ← Only z10 or older <ul style="list-style-type: none"> • Copper cable plugs close to memory bus • 10 meter max length
Extended Distance	<ul style="list-style-type: none"> • 12x IFB, 12x IFB3 <ul style="list-style-type: none"> • 150 meter max distance optical cabling • ISC-3 (InterSystem Channel) <ul style="list-style-type: none"> • Fiber optics • I/O Adapter card • 10km and longer distances with qualified DWDM solutions • 1x IFB <ul style="list-style-type: none"> • Fiber optics – uses same cabling as ISC • 10km and longer distances with qualified DWDM solutions

11


IC links are Licensed Internal Code-defined links to connect a CF to a z/OS logical partition in the same server. These links are available on all System z servers. The IC link is a System z server coupling connectivity option that enables high-speed, efficient communication between a CF partition and one or more z/OS logical partitions running on the same server. The IC is a linkless connection (implemented in Licensed Internal Code) and so does not require any hardware or cabling.

ISC-3 links provide the connectivity required for data sharing between the CF and the systems. ISC links support point-to-point connections (directly connecting CFs and systems), and require a unique channel definition at each end of the link.

InfiniBand coupling links (PSIFB) are high speed links on z196, z114, z10 and z9 servers. PSIFB coupling links use a fiber optic cable that is connected to a Host Channel Adapter fanout in the server.

Integrated Cluster Bus links are members of the family of coupling link options available on System z10 and previous System z servers. They are faster than ISC links, attaching directly to a Self-Timed Interconnect (STI) bus of the server. The ICB features are highly integrated, with very few components, and provide better coupling efficiency (less server overhead associated with coupling systems) than ISC-3 links. They are an available method for coupling connectivity when connecting System z10 and previous System z servers over short distances (seven meters).

The choice of coupling link technology is dictated by several factors:

- Machine types. Not all machines support all types of links. Different machines have different limits on the number of links that can be configured.
- Distance. Each link technology has a maximum supported distance.
- Speed. The speed of the links directly impacts the response time for CF requests.

InfiniBand (IFB) Overview



- Physical lane
 - Link based on a two-fiber 2.5 Gbps bidirectional connection for an optical (fiber cable) implementation
 - Grouped as either 12 physical lanes (12x) or 1 physical lane (1x)
- Link speeds
 - Single data rate (SDR) delivering 2.5 Gbps per physical lane
 - Double data rate (DDR) delivering 6.0 Gbps per physical lane
- Host Channel Adapter (HCA)
 - Physical devices that create and receive packets of information
- CHPID type (CIB) for both 12x IFB and 1x IFB
 - 7 subchannels per CHPID (12x)
 - 32 for HCA2-O LR and HCA3-O LR (1x)
- IFB available on z196, z114, z10, z9
 - HCA3-O exclusive to z196 and z114

12



Infiniband links are sometimes called IFB, PSIB, PSIFB, IB Coupling, or CIB (coupling over infiniband). They are all the same thing.

There are several ways of classifying IFB links. It can be Double Data Rate (DDR) or Single Data Rate (SDR). The IFB links can also be characterized by the number of lanes of communication within each physical link. For example, 12x links provide 12 lanes of communication within each physical link. 1x links provide a single lane of communication within each physical link.

7 subchannels per CHPID is same as for other links (ISC, ICB). What's unique to infiniband is ability for the physical link to share CHPIDs.

On July 12, 2011 IBM announced that 1x IFB links connecting a z196 to either a z196 or a z114 via HCA2-O LR or HCA3-O LR can have 32 subchannels per CHPID. HCD will define 32 subchannels per CHPID by default. The installation can optionally have HCD define 7 subchannels as in the past. If you choose 32 subchannels for a connection on a short 12x link, you will be wasting system resources by providing a more subchannels than are actually necessary (and more than can actually be used, since there are only 7 actual link buffers); excessive path busy conditions may result due to this over-commitment of subchannels relative to link buffers. If you choose 7 subchannels for a connection on a long 1x link, it may not be possible for you to fully utilize the capacity of the link (since there are 32 actual link buffers available for use), which may unnecessarily constrain the maximum CF command throughput that can be achieved on the link.

Note: The InfiniBand link data rates of 6 GBps, 3 GBps, 2.5 Gbps, or 5 Gbps do not represent the performance of the link. The actual performance is dependent upon many factors including latency through the adapters, cable lengths, and the type of workload.

New Infiniband Support

July 12, 2011



- Improved service times with 12x IFB links
 - New 12x IFB3 protocol, applies when:
 - HCA3-O to HCA3-O connectivity, ≤ 4 CHPIDs/port
 - Designed to improve link service time up to 40%
- Improved physical connectivity with 1x IFB links
 - HCA3-O LR has 4 ports instead of 2
 - Helps with migration from ISC-3 links
- Up to 32 subchannels per CHPID with 1x IFB links
 - HCA3-O LR or HCA2-O LR
 - Helps with bandwidth at longer distances

13



New infiniband support was announced July 12, 2011.

Improved service times when using the 12x IFB3 protocol: The HCA3-O feature supports two communication protocols: 12x IFB and 12x IFB3.

12x IFB3 protocol: When HCA3-Os are communicating with HCA3-Os and have been defined with four or fewer CHPIDs per port, the 12x IFB3 protocol is utilized. When using this protocol, synchronous service times are designed to be 40% faster. The service time for CF requests should be faster as well, but not by 40% since the link service time is only one component of the service time for a CF request.


12x IFB protocol: If more than four CHPIDs are defined per HCA3-O port, or HCA3-O features are communicating with HCA2-O features on zEnterprise or System z10 servers, links will run with the 12x IFB protocol.

Improved physical connectivity with 1x InfiniBand coupling links: The HCA3-O long reach (LR) fanout for 1x InfiniBand coupling links has been designed with four ports per feature to satisfy requirements for more physical connectivity. The added connectivity will be helpful as clients migrate from InterSystem Channel-3 (ISC-3) to 1x InfiniBand coupling links.

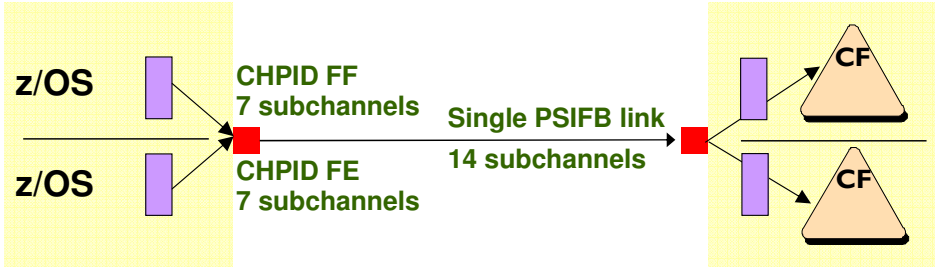
Up to 32 subchannels (devices) per CHPID for 1x InfiniBand coupling links: To improve link utilization and coupling throughput at extended distances between the coupling facility (CF) and the operating system or between CFs separated by extended distances, we are now optionally supporting 32 subchannels (devices) per CHPID, versus the current 7 subchannels per CHPID. The additional subchannel definition is available whether using an HCA3-O LR or HCA2-O LR feature for 1x IFB. This increase is designed to help eliminate the need to add CHPIDs or physical links in order to help improve coupling throughput at extended distances.

The 12x InfiniBand HCA3-O fanout can connect to 12x InfiniBand HCA2-O fanout on a z196, z114, or z10. An HCA3-O fanout cannot connect to an HCA1-O fanout on a z9™


IFB Supports Multiple Channel Paths per Physical Link



- Allows more subchannels per physical link
 - Up to 16 CHPIDs across the ports of a single InfiniBand coupling HCA
- Can connect to multiple CF LPARs



- MIF uses same address, 7 subchannels per CHPID
- With HCA2-O LR or HCA3-O LR, 32 subchannels per CHPID



14

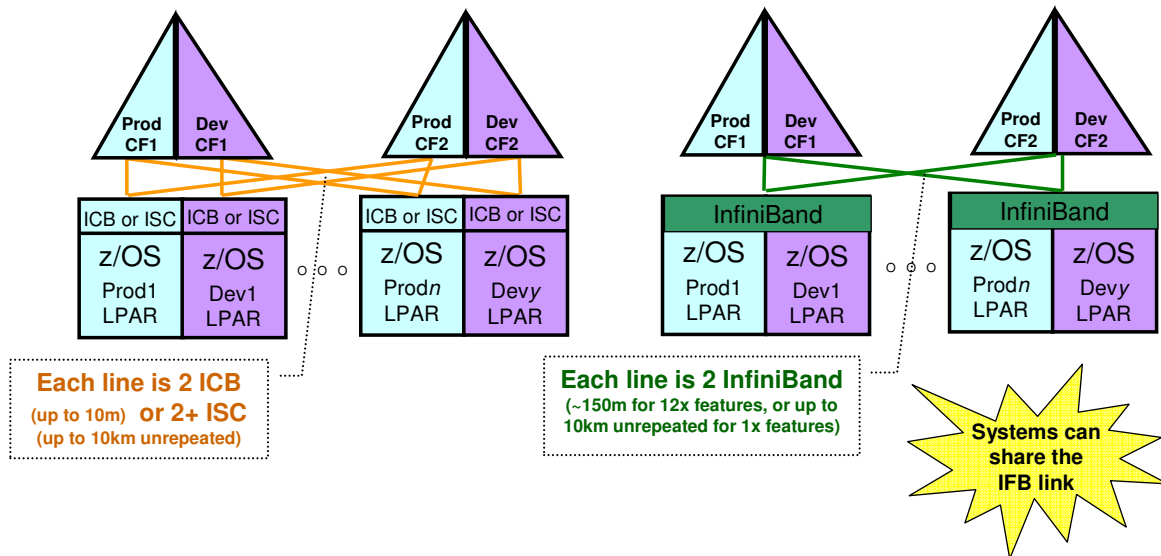
From a capacity perspective, it is unlikely that one would need more than 8 CHPIDs per IFB link. For heavy loads, configuring more than 8 CHPIDs across the multiple ports of an InfiniBand coupling HCA could in fact hurt performance as there comes a point of diminishing returns. With light loads, one might configure more than 8 CHPIDs for connectivity reasons. If you think you need to go beyond 8, it would be wise to carefully review your particular situation with advanced technical support.

Subchannel Busy Conditions: One might configure additional links to a CF in order to mitigate “subchannel busy” conditions that can occur when a z/OS system has a spike in CF requests. Lack of subchannel causes delay. z/OS may “spin” the processor until a subchannel becomes available, or it may have to queue the request. Prior to infiniband links, one would have to configure a new physical link in order to provide the system with additional subchannels. With infiniband links, one can simply configure an additional CHPID for the link (within the supported limits).

Path Busy Conditions: When two z/OS images share a physical link, each individual image may appear to have an available subchannel. However, when the CF request gets to the link, the link may report a “path busy” condition. This condition also leads to delay because z/OS must now redrive the request. Prior to infiniband links, one would have to configure a new physical link to avoid the problem. With infiniband links, one can simply configure an additional CHPID for the link (within the supported limits).

Connect to multiple CF images: Prior to infiniband links, the physical link had to be dedicated to a particular target CF image. So if an installation configured two z/OS images in different sysplexes on CEC1, and two CF images on CEC2 with one CF for each sysplex, two physical cables (plus 2 for redundancy) would have been required to connect the two CECs. With infiniband links, one physical infiniband cable (plus 1 for redundancy) can be used to connect the two CECs. The physical cable can be share by the two sysplexes, but each system’s CF requests will be directed to the CF image being used by the relevant sysplex. The ability to connect to multiple CF images can greatly simplify the physical configuration.

Lower Cost Coupling Infrastructure – consolidating coupling links



15

Infiniband links can be used to connect to multiple CF images. Doing so can simplify the configuration and reduce cost. The chart illustrates how two sysplexes, one for production work and one for test, might be spread across multiple CECs. We envision two external coupling facilities, each containing two CF images, one for each sysplex. On the left side, the configuration requires a unique cable for each target CF image (plus one more for redundancy) per sysplex per CEC. On the right, the same logical connectivity can be achieved by an infiniband link (plus one more for redundancy) run between the CECs and the external CF.

Consolidating links with IFB



- **Pure Capacity**
 - 1 12x IFB replaces 1 ICB-4
 - 1 12x IFB replaces 4 ISC-3s
- **Eliminating subchannel and path delays**
 - Often >2 ICB-4s configured not for capacity but for extra subchannels/paths to eliminate delays
 - 2 12x IFB links with multiple CHPIDs can replace >2 ICB-4s in this case
- **Multiple sysplexes sharing hardware**
 - Production, development, test sysplexes may share hardware
 - Previously each required own ICB-4 or ISC-3 links
 - 2 12x or 1x IFB links with multiple CHPIDs can replace >2 ICB-4s or ISC-3s in this case
- **Multiple CHPID recommendations**
 - Max 16 per HCA (2 ports per HCA)
 - Use up to all 16 for lightly loaded connectivity
 - Limit to max of 8 per HCA for heavy loads

Be sure to maintain redundancy !

16



From a pure capacity perspective, one 12x PSIFB link is equivalent to either one ICB4 link or four ISC3 links.

Many installations configuring ICB4 links for their z9 or z10 servers, find that two such links between the z/OS image and the CF are sufficient. Some installations need the added capacity of a 3rd or 4th ICB. In those cases, an installation looking to replace ICB4 links with PSIFB links would do so on a one for one basis. More typically, installations need to configure a 3rd or 4th ICB4 link not for capacity reasons, but to overcome delays caused by “path busy” and “subchannel busy” conditions that can occur due to “bursty” traffic. In those cases, two PSIFB links with a pair of CHPIDs assigned to each link could replace the three or four ICB4 links. That is, one PSIFB link could replace more than one ICB4 link.

Some installations have multiple CECs, with multiple sysplexes running on each CEC. So on CEC “B”, one might for example have coupling facility CF1B used by sysplex 1, and coupling facility CF2B used by sysplex 2. CEC “A” would need at least two ICB4 links (for example) to allow the z/OS images on CEC “A” to communicate with those coupling facilities. One link would connect CEC “A” to coupling facility CF1B, and the other would connect it to CF2B. A separate link would be needed for each CF because there is a one-to-one correspondence between a CF receiver CHPID and an ICB4 link. However, one PSIFB link could be used to connect the two CECs because the receiver CHPIDs for each CF on CEC “B” could share the PSIFB link. In these cases, where multiple ICB4 and ISC3 links are configured to establish necessary connectivity (as opposed to capacity), one PSIFB link with multiple CHPIDs could be used to replace multiple links.

Some real world examples where installations have converted to PSIFB links.

-Consolidating to System z10 model E64 which does not support ICB-4.

- Consolidation of 16 ISC-3 links to 4 IFB (maintaining redundancy across two HCA) within a datacenter. Infrastructure savings and improvement in coupling efficiency.

-Installation had 2 ICB for development and 4 ICB for production Sysplex sharing System z10. Converted to 4 PSIFB links with shared CHPIDs for development and production without a significant decrease in coupling efficiency.

After PSIFB links are configured, additional CHPIDs can be added to a configuration providing additional subchannels without requiring additional physical hardware infrastructure.

Coupling Technology versus Host Processor Speed

Host effect with primary application involved in data sharing

Chart below is based on 9 CF ops/Mi - may be scaled linearly for other rates

Host CF	z9 BC	z9 EC	z10 BC	z10 EC	z114	z196
z9 BC ISC3	14%	15%	17%	19%	18%	23%
z9 BC 12x IFB	NA	NA	13%	14%	13%	16%
z9 BC ICB4	9%	10%	11%	12%	NA	NA
z9 EC ISC3	13%	14%	16%	18%	17%	22%
z9 EC 12x IFB	NA	NA	13%	14%	13%	16%
z9 EC ICB4	8%	9%	10%	11%	NA	NA
z10 BC ISC3	13%	14%	16%	18%	17%	22%
z10 BC 12x IFB	11%	12%	13%	14%	13%	15%
z10 BC ICB4	8%	9%	10%	11%	NA	NA
z10 EC ISC3	12%	13%	15%	17%	17%	22%
z10 EC 12x IFB	10%	11%	12%	13%	12%	15%
z10 EC ICB4	7%	8%	9%	10%	NA	NA
z114 ISC3	14%	14%	16%	18%	17%	21%
z114 12x IFB	10%	10%	12%	13%	12%	15%
z114 12x IFB3	NA	NA	NA	NA	10%	12%
z196 ISC3	11%	12%	14%	16%	17%	21%
z196 12x IFB	9%	10%	11%	12%	11%	14%
z196 12x IFB3	NA	NA	NA	NA	9%	11%

With z/OS 1.2 and above, synch->asynch conversion caps values in table at about 18%
 PSIFB 1X links would fall approximately halfway between PSIFB 12X and ISC links
 IC links scale with speed of host technology and would provide an 8% effect in each case

17

The coupling efficiency of a Parallel Sysplex cluster, particularly one that has heavy datasharing, is sensitive to the performance of the operations to the Coupling Facility. The chart estimates the "host effect" for a heavy data sharing production workload for various combinations of host processor and coupling technology. The values in the table represent the percentage of host capacity that is used to process operations to the coupling facility (which includes the time spent communicating with the CF and the time spent waiting for the CF to process the request). For example, a value of 10% would indicate that approximately 10% of the host capacity (or host MIPS) is consumed by the subsystem, operating system and hardware functions associated with coupling facility activity. The table is based on a "coupling intensity" of 9 CF operations per million instructions (MI), which is typical of high end data sharing work loads.

The values in the table can be adjusted to reflect the coupling intensity for any workload. One can calculate the coupling intensity by simply summing the total req/sec of the CFs and dividing by the used MIPS of the attached systems (MIPS rating times CPU busy). Then, the values in the table would be linearly scaled. For example, if the workload was processing 4.5 CF operations per million instructions (or 4.5 CF ops/second/MIPS), then all the values in the table would be cut in half.

For 9 CF requests/MI, host effect values in the table may be considered capped at approximately 18% due to z/OS Synchronous to Asynchronous CF Message Conversion. Configurations where entries are approaching 18% will see more messages converted to asynchronous. z/OS converts synchronous messages to asynchronous messages when the synchronous service time relative to the speed of the host processor exceeds a breakeven threshold at which it becomes cheaper to go asynchronous. When all CF operations are asynchronous, the overhead will be about 18%. By the time you have reached $\geq 18\%$ in the table, that corresponds to the time z/OS must have been converting almost every operation asynchronous. The 18% cap scales proportionally with the CF requests/MI activity rate. For example, at 4.5 CF requests/MI, the cap would be 9%.

The hardware cost can be minimized by using the most efficient links with faster engine speeds for the CFs. This reduces the time that z/OS is waiting for the response while the message is on the CF link and while the CF is busy processing the request. With this in mind, it becomes obvious that the best coupling efficiency is generally seen when the CF is on the fastest processor and connected to the z/OS image via the fastest links. The chart bears this out. For example, holding the CF and host processor technology constant, the chart shows that coupling efficiency increases with faster links (z/OS spends less time waiting because the communication with the CF is faster). For a given host processor and link technology, coupling efficiency increases with faster CF technology (z/OS spends less time waiting because the CF processes the request faster). In most cases, upgrading to faster links has a more dramatic impact on coupling efficiency than upgrading to a faster CF.

Maximum Coupling Links and CHPIDs



Server	1x IFB (HCA3-O LR)	12x IFB 12x IFB3 (HCA3-O)	1x IFB (HCA2-O LR)	12x IFB (HCA2-O)	IC	ICB-4	ICB-3	ISC-3	Max External Links	Max Coupling CHPIDs
z196	48 M15 – 32*	32 M15 – 16*	32 M15 – 16*	32 M15 – 16*	32	N/A	N/A	48	104 ¹	128
z114	M10 – 32* M05 – 16*	M10 – 16* M05 – 8*	M10 – 12 M05 – 8*	M10 – 16* M05 – 8*	32	N/A	N/A	48	M10 ² M05 ³	128
z10 EC	N/A	N/A	32 E12 - 16	32 E12 - 16	32	16 ⁴	N/A	48	64	64
z10 BC	N/A	N/A	12	12	32	12	N/A	48	64	64
z9 EC	N/A	N/A	N/A	16 S08 - 12	32	16	16	48	64	64
z9 BC	N/A	N/A	N/A	12	32	16	16	48	64	64

1. z196 & z114 do not have an inherent maximum external link limit. The effective limit depends on the HCA fanout slots available and combined 12x IFB and 1x IFB limit of 16 HCA features
z196 M49, M66 or M80 supports max 96 extended distance links (48 1x IFB and 48 ISC-3) plus 8 12x IFB links
z196 M32 supports max 96 extended distance links (48 1x IFB and 48 ISC-3) plus 4 12x IFB links*
z196 M15 supports max 72 extended distance links (24 1x IFB and 48 ISC-3) with no 12x IFB links*
2. z114 M10 supports max 72 extended distance links (24 1x IFB and 48 ISC-3) with no 12x IFB links*
3. z114 M05 supports a maximum of 56 extended distance links (8 1x IFB and 48 ISC-3) with no 12x IFB links*
4. ICB-4 not supported on Model E64. 32 ICB-4 links with RPQ on z10 EC

* Uses all available fanout slots. Allows no other I/O or coupling
18



Max external links is the maximum total number of physical link ports (does not include IC)

Max coupling CHPIDs defined in IOCDs includes IC and multiple CHPIDs defined for IFB.

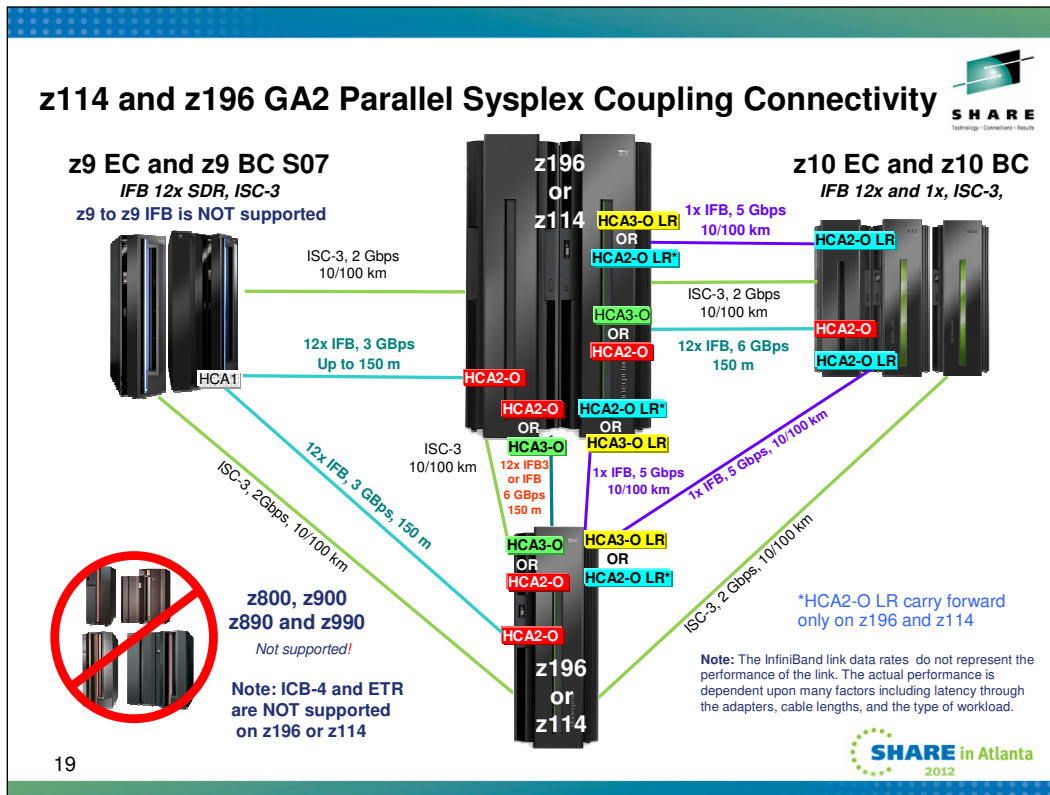
When originally introduced, the zEnterprise 196 increased the number of external coupling links from 64 to 80, which allowed the full configuration of 32 IFB links and 48 ISC-3 links to be used. With the new HCA3-O LR, which support 4 ports per HCA, the z196 can now support up to 104 external links.

With z196 and z114, the number of coupling CHPIDs per server has been increased from 64 to 128.

A z196 supports up to a maximum of 16 HCA fanouts.

A z114 has a maximum of four (4) HCA-O fanouts on the M05 and eight (8) HCA-O fanouts on M10

The I/O fanouts compete for fanout slots used by the infiniband HCA fanouts. So some of the maximums listed in the chart are likely not achievable in a “real” configuration as some of the slots would be needed for traditional I/O fanouts, which in turn would reduce the number of HCA fanouts below what is needed to achieve the maximum. The number and type of HCA fanouts installed determines the number of InfiniBand links that can be configured.



19

This chart summarizes the various types of coupling links that can be used to connect systems in a sysplex that includes z196 and/or z114 servers. Each line indicates the type of link, the link speed, and distance limitations. For infiniband links, the endpoints of the line indicate the HCA that can be used.

Note:

1. IBM does not support InfiniBand coupling links for System z9-to-System z9 communication. InfiniBand coupling links on System z9 are intended to support a migration to System z10, z114, or z196.
2. The z196 and z114 servers cannot connect to a z800, z900, z890 or z990 in a Parallel Sysplex environment.

The 12x InfiniBand HCA3-O fanout can connect to the 12x InfiniBand HCA2-O fanout on a z114, z196, or z10, but an HCA3-O fanout cannot connect to an HCA1-O fanout on a z9.

Infiniband Resource Materials



- Parallel Sysplex Website
 - www.ibm.com/systems/z/advantages/pso/ifb.html
 - “*Coupling Facility Configuration Options*” whitepaper
- Redbooks
 - "Implementing and Managing InfiniBand Coupling Links on System z", SG24-7539
 - “IBM System z Connectivity Handbook” SG24-5444
- STG Lab Services
 - Specialized studies available for complex situations
 - Send a note to stgls@us.ibm.com
- zCP3000 (**Performance Analysis and Capacity Planning**)
 - Includes reports of CF and CP utilization given a change of coupling link types
 - Contact your IBM representative to use this tool

20



Redbooks are available at www.redbooks.ibm.com

An older version was titled “*Getting Started with InfiniBand on System z10 and System z9*” SG24-7539


The “*Coupling Facility Configuration Options*” is available at <http://www.ibm.com/systems/z/advantages/pso/whitepaper.html>

www.ibm.com/support/techdocs is another source for whitepapers and other technical information.

Agenda



- Hardware Updates
 - CFCC Level 17
 - InfiniBand (IFB) coupling links
 - [Server Time Protocol \(STP\)](#)
- Software Updates
 - z/OS V1R13
 - z/OS V1R12
 - z/OS V1R11
- Summary

Glossary for Server Time Protocol (STP) 		
Acronym	Full name	Comments
Arbiter	Arbiter	Server assigned by the customer to provide additional means for the Backup Time Server to determine whether it should take over as the Current Time Server.
BTS	Backup Time Server	Server assigned by the customer to take over as the Current Time Server (stratum 1 server) because of a planned or unplanned reconfiguration.
CST	Coordinated Server Time	The Coordinated Server Time in a CTN represents the time for the CTN. CST is determined at each server in the CTN.
CTN	Coordinated Timing Network	A network that contains a collection of servers that are time synchronized to CST.
CTN ID	Coordinated Timing Network Identifier	Identifier that is used to indicate whether the server has been configured to be part of a CTN and, if so, identifies that CTN.
CTS	Current Time Server	A server that is currently the clock source for an STP-only CTN.
PTS	Preferred Time Server	The server assigned by the customer to be the preferred stratum 1 server in an STP-only CTN.

STP - Time synchronization for Parallel Sysplex

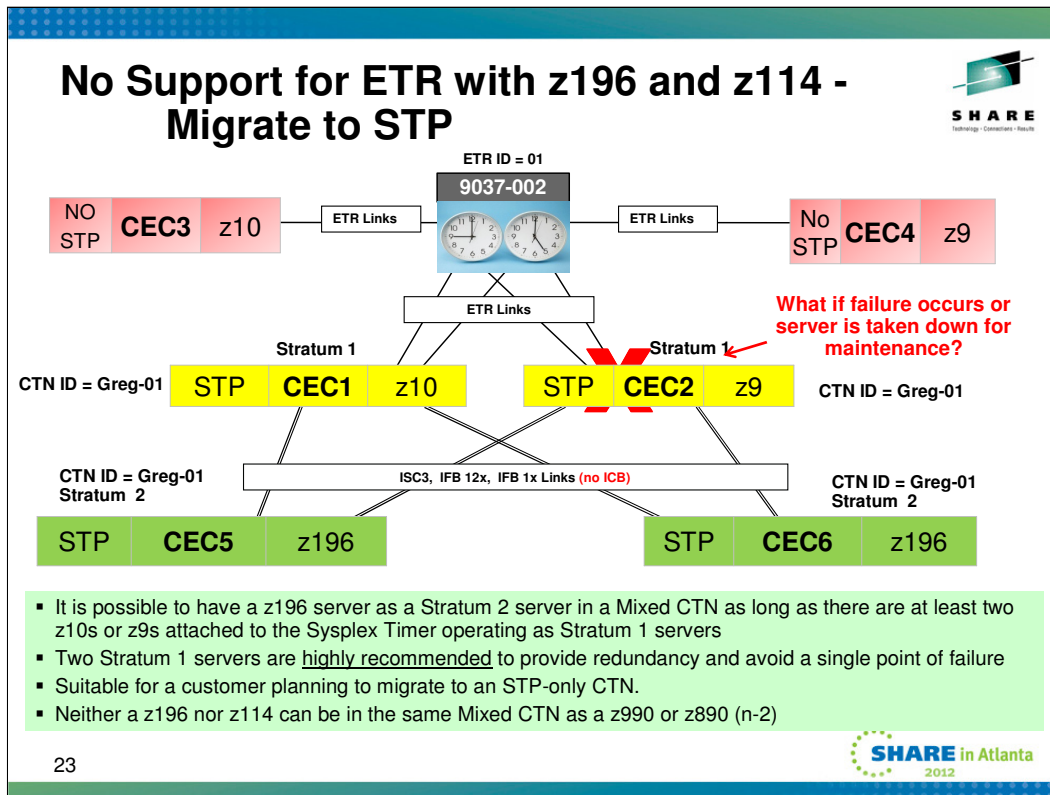
Server Time Protocol (STP) is designed to allow events occurring in different servers to be properly sequenced in time. STP is designed for servers that have been configured in a Parallel Sysplex or a basic sysplex (without a coupling facility), as well as servers that are not in a sysplex but need time synchronization.

STP is a server-wide facility that is implemented in the Licensed Internal Code (LIC), presenting a single view of time to Processor Resource/Systems Manager™ (PR/SM). STP uses a message-based protocol in which timekeeping information is passed over externally defined coupling links between servers. The coupling links that can be used to transport STP messages include 12x or 1x InfiniBand (IFB) links and InterSystem Channel-3 (ISC-3) links configured in peer mode. These can be the same links that already are being used in a Parallel Sysplex for coupling facility (CF) message communication. STP can also use ICB-3 and ICB-4 links on servers that support them.

The STP design introduced a concept called Coordinated Timing Network (CTN). A Coordinated Timing Network (CTN) is a collection of servers and coupling facilities that are time-synchronized to a time value called Coordinated Server Time.

A CTN can be configured in two ways:

- STP-only CTN which does not require a Sysplex Timer®.
- Mixed CTN (External Time Reference (ETR) and STP) which requires a Sysplex Timer. The Sysplex Timer provides the timekeeping information in a Mixed CTN. Even though the z196 and z114 do not support attachment to a Sysplex Timer, they can participate in a Mixed CTN that has either a z10 or z9 synchronized to the Sysplex Timer. This maintains the capability for enterprises to concurrently migrate from an existing ETR network to a Mixed CTN and from a Mixed CTN to an STP-only CTN.



Hopefully you have already completed your migration to STP. This chart illustrates a mixed CTN (External Time Reference (ETR) and STP) which uses a Sysplex Timer. The Sysplex Timer provides the timekeeping information in a Mixed CTN. Even though the z196 and z114 do not support attachment to a Sysplex Timer, they can participate in a Mixed CTN that has either a z10 or z9 synchronized to the Sysplex Timer. This maintains the capability for enterprises to concurrently migrate from an existing ETR network to a Mixed CTN and from a Mixed CTN to an STP-only CTN.

Helpful Redbooks available at www.redbooks.ibm.com:

Server Time Protocol Planning Guide, SG24-7280

Server Time Protocol Implementation Guide, SG24-7281

Server Time Protocol Recovery Guide, SG24-7380

Also, if you are configuring or have already configured a Server Time Protocol (STP) Coordinated Timing Network (CTN) with three or more servers, see the white paper “**Important Considerations for STP server role assignments**” (WP101833), for some important tips to help prevent a sysplex wide outage. The paper is available at www.ibm.com/support/techdocs.

Background: Support Element (SE) Battery Operated Clock (BOC)



- SE Clock used to set CPC clock at IML
- SE Clock synchronized to CPC clock
- NTP clients on zBX synchronize to SE Clock

- But BOC synchronized only once every 24 hours:
 - So CPC TOD after IML could have significant drift relative to External Time Source of CPC – could take hours to steer TOD back in sync, and
 - Time coordination between z196 and zBX components typically worse than non-System z servers not part of the zEnterprise, and
 - Synchronization by “jumping” implies timestamp anomalies

24



The Support Element (SE) Battery Operated Clock (BOC) is used to set the CPC TOD at IML time. After IML, the primary SE synchronizes its clock with the CPC TOD. The oscillators of the SE clock are not as stable as the CPC clock and so may have significantly more drift. Furthermore, the CPC and other servers throughout the enterprise are likely to be synchronized to an External Time Source (ETS). Thus synchronizing to the CPC clock is the way to go.

As for all the servers in the enterprise, it is desirable for the components of the zBX to have a common time source. NTP clients on the zBX firmware components access the NTP server on the SE every 2 to 4 hours to accomplish this.

Problems (apply to zEnterprise pre-driver 93, e.g. z196 GA1)

- The SE BOC can accumulate a significant amount of drift in 24 hours. Thus if customer has used option to Save STP configuration in a single or dual server CTN, after a power off/on sequence, the SE's inaccurate clock is used to set the CEC TOD. It may take several hours or longer to steer the CEC TOD to within 100 ms of the configured External Time Source (ETS).
- Time coordination between z196 and zBX components is typically worse than what a customer can achieve between z196 and non-System z servers that are not part of the zEnterprise. Having the zBX components in zEnterprise synch with SE BOC every 2 to 4 hours is really only good enough to maintain reasonable time consistency for logging purposes. Furthermore, the zBX is being synchronized to a clock that is likely drifting significantly from the ETS that the other servers in the enterprise are using.
- Synchronization of the SE BOC is accomplished by setting the BOC equal to the current CPC TOD. Thus time on the SE can “jump”, either forwards or backwards. Jumping backwards can cause duplicate time stamps. Jumping forwards can cause gaps in time which can, for example, make elapsed time appear larger than it actually was.

Improved SE BOC Synchronization



Applies to z196 GA2 and z114 (driver 93)

- SE synchronizes to CPC TOD every hour
- Synchronization is steered

So:

- CPC TOD set at IML from SE BOC has less drift
- SE BOC can maintain time accuracy within 100 milliseconds of CPC (and therefore ETS)
- NTP Clients on zBX that synchronize with SE at least once an hour can maintain it too

25



The improvements apply to z196 or z114 (driver 93) independent of whether zBX or STP is configured. The Support Element (SE) will synchronize its Battery Operated Clock (BOC) to the CPC TOD every hour instead of every 24 hours. The SE accomplishes the synchronization by steering the BOC TOD instead of setting (jumping) the time to make the adjustments required to synch with the CEC TOD. These enhancements allow the SE's clock to maintain a time accuracy of 100 milliseconds to an NTP server configured as the External Time Source in an STP-only CTN.

With the new support, we get **Improved time coordination for zBX components**: Network Time Protocol (NTP) clients, running on blades in the zBX, can synchronize their time to the NTP server provided by the Support Element (SE) every hour. This enhancement provides the capability for the components in the zBX to maintain an approximate time accuracy of 100 milliseconds to an NTP server if they synchronize to the SE's NTP server at least once an hour. This enhancement is exclusive to the z196 and z114.

Background: STP only CTN Recovery rules for 3 or more servers



To ensure data integrity, you CANNOT have two Stratum 1 servers in timing network

- Backup Time Server (BTS) can take over as Current Time Server (CTS), if:
 - Preferred Time Server (PTS) can indicate it has “failed”, or
 - BTS can unambiguously determine the PTS has “failed”
- Otherwise recovery based on “voting” (2 out of 3)
 - If BTS and Arbiter agree they cannot communicate with PTS, then safe for BTS to takeover as S1
 - If PTS cannot communicate with BTS and Arbiter, it can no longer remain a S1 (even if it is operational and providing a time source to rest of CTN)

26



In order to keep the complex discussion that follows as understandable as possible, it will be assumed that PTS has been assigned as the CTS (Active Stratum 1), BTS is a Stratum 2 server capable of becoming the CTS.

Note that you can have only one CTS in the CTN, and it can be either the PTS or BTS.

When you plan to take a disruptive action against a server that has a CTN role, you should reassign the CTN role prior to taking the disruptive action. When a server with a CTN role is no longer capable of performing that role, the role should be reassigned. Your operational procedures must be updated to account for the need to reassign these roles. Failure to reassign the role can have dire consequences (such as a sysplex wide outage). **This is important!**

If the PTS, BTS or Arbiter is no longer an operational synchronized server in the CTN, perform one of the following actions to maintain a resilient STP-only CTN:

- Reassign PTS, BTS, Arbiter roles if connectivity permits, or
- Reassign only the PTS and BTS roles if either: no server available to assign as Arbiter, or connectivity does not permit any server to be an Arbiter
- Reassign only the PTS as the CTS if no other server can perform the role of the BTS

The assignment of roles in a CTN is typically done with great care as there are many things that need to be considered to ensure that the configuration will meet the needs of the enterprise in the face of various failures. So in general, **after a failure condition is repaired, you will want to reassign the PTS, BTS, Arbiter roles to restore the normal operational state of CTN.**

Motivational Example: Sysplex Outage 1 IBM

CTN roles not reassigned prior to planned disruptive actions on special role servers

- Planned disruptive action initiated from HMC to POR BTS and Arbiter
 - Could be same task or
 - Could be sequential
- **Roles not reassigned as recommended**
- PTS loses communication to both BTS and Arbiter
 - Surrenders S1 role
 - Since no other clock source available, PTS becomes unsynchronized (S0)
 - CECs with P4, P5, P6 also become unsynchronized
- **Sysplex wide outage**

27 © 2012 IBM Corporation

In this example, we have a Server Time Protocol (STP) only Coordinated Time Network (CTN) with one server defined as the Preferred Time Server (PTS), another as the Backup Time Server (BTS), and another as the Arbiter. The customer intends to take a disruptive action that will take down both the BTS and the arbiter. However, the customer does not reassign the CTN roles before initiating the action. When the BTS and arbiter go down, the PTS is unable to communicate with those servers. The triad voting rules require the PTS to surrender its role as the CTS when it is unable to communicate with both the BTS and the Arbiter. When the PTS surrenders its role as CTS, none of the systems in the sysplex have a clock source. They all become unsynchronized and a sysplex wide outage ensues.

To avoid the outage, the customer should have reassigned the roles. Ideally, two other servers in the sysplex would be assigned the BTS and Arbiter roles. But the sysplex depicted in the slide does not have the connectivity needed to allow both roles to be assigned. So for this sysplex, prior to taking the disruptive action against P2 and P3, one would either eliminate both the BTS and Arbiter roles completely, or assign the BTS role to one of the servers P4, P5, or P6. Deleting both roles implies that a subsequent failure of the PTS would cause a sysplex wide outage. Assigning the BTS role to P4 (for example), would avoid the sysplex wide outage as P4 would survive the failure of the PTS. P4 would be the only surviving system, but that is better than losing the entire sysplex.

At any rate, reassigning the roles to other servers prior to taking the disruptive action against P2 and P3 eliminates that sysplex wide outage. If both roles are reassigned, taking down P2 and P3 becomes irrelevant to the CTN. If both roles are eliminated, or if only the BTS role is assigned, the triad voting rules would not be applied.

However, note that the system prevents disruptive actions from being taken against the PTS. In this example, the disruptive actions against the BTS and Arbiter are dangerous. Wouldn't it be nice if the system helped as avoid the sysplex outage by preventing these disruptive actions as well?

Block disruptive planned actions



Applies to z196 GA2, z196 GA1 w/ MCLs, z114, and z10 w/ MCLs

- Block planned disruptive action from proceeding on the BTS and Arbiter using similar design used to block CTS
- Forces CTN role to be removed or reassigned prior to proceeding with a disruptive action
- Pros:
 - Safeguard for disruptive planned actions performed as part of same HMC task on both the BTS and Arbiter or done sequentially
 - Consistency – same procedure for all servers with CTN roles
- Cons:
 - Need special operational procedures for each server assigned a role in the CTN, instead of just the CTS
 - Does not safeguard against unplanned double failures

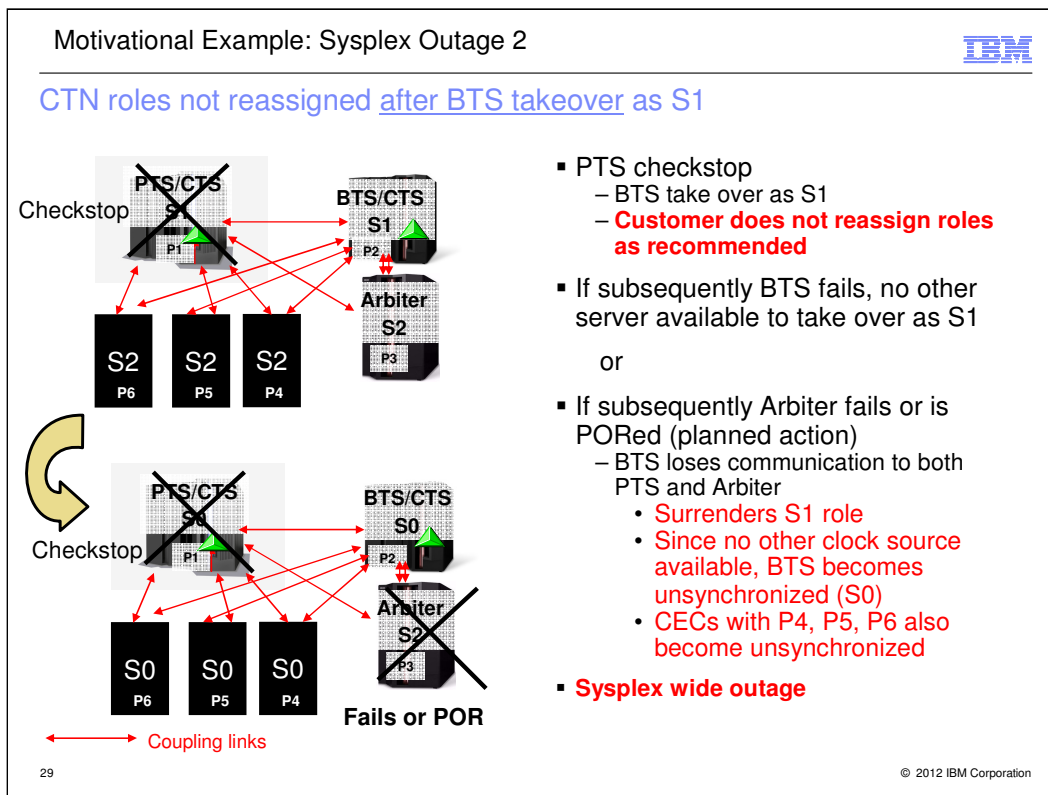
28



With STP, the SE has safeguards to block a planned disruptive operation against the Preferred Time Server (PTS), assuming it is the CTS, if it had initialized coupling links to at least one Stratum 2 server in the CTN. Examples of disruptive operations are activate, deactivate, power off, power-on reset, disruptive switch to alternate SE. The disruptive action would not be permitted to proceed until the role of PTS have been reassigned or all the CHPIDs were configured offline. The intent is to prevent a planned disruptive action that could lead to a loss of a time server in the sysplex, which in turn could cause one or more (possibly all) of the systems in the sysplex to fail.

Prior to this enhancement, disruptive actions were permitted against the BTS and the arbiter. The primary problem with this behavior is that it exposes the installation to potential sysplex wide outages. Blocking disruptive actions against any server that has a CTN role helps avoid unexpected outages.

Driver/Server	MCL	Bundle	Release Date
D86E / z196	N29809.277	45	Sept 8, 2011
D86E / z196	N29802.420	45	“
D79F / z10	N24415.078	50	Sept 28, 2011
D79F / z10	N24409.184	50	“
D93G / z114	Integrated	N/A	Sept 9, 2011
D93G / z196 GA2	Integrated	N/A	Sept 9, 2011



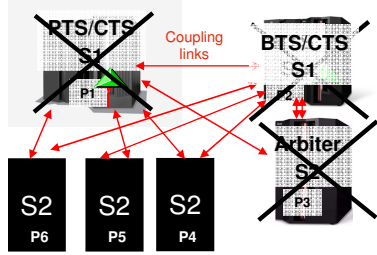
In this example, when the Preferred Time Server (PTS) fails, the Backup Time Server (BTS) and the Arbiter agree that the PTS is gone. Thus the BTS takes over as Current Time Server (CTS). At this point, we now have the BTS as a single point of failure for the sysplex as it is the only time source. If the BTS fails, the sysplex loses time synchronization and we suffer a sysplex wide outage.


However, under the assumption that the CTN roles are not reassigned, we have a second single point of failure as well. The arbiter! If the arbiter subsequently fails, we will then have a situation where the BTS has lost communication with both the PTS (it still being down) and the arbiter. Since all three roles are still assigned, the triad voting rules apply. These rules require the BTS to surrender its role as the CTS if it cannot communicate with the PTS and Arbiter. Since there is no other clock source, the BTS becomes unsynchronized. Indeed, the remaining systems in the sysplex become unsynchronized as well because they have no CTS. Once again we have a sysplex wide outage, one which is perhaps surprising in that the arbiter did not have any direct role in providing time and the BTS was otherwise fully operational, capable of providing time signals.

The sysplex outage due to the BTS failing after it takes over as CTS from the failed PTS is perhaps understandable (at least for this configuration). However, it should be noted that reassigning the CTN roles with for example, P2 as PTS and P4 as BTS would have allowed the sysplex to survive a subsequent failure of P2 (albeit with just one system).

But the second scenario, where the failure of the arbiter P3 after the BTS takes over as CTS from the failed PTS leads to a sysplex outage, is at best surprising. It seems that strict adherence to the triad voting rules produced catastrophic results. We really ought to do better.

STP design changes






Triad voting disabled when Triad state “degraded”

- Triad State “degraded” when any two of the special role servers (PTS, BTS, Arbiter) agree they cannot communicate with the third special role server
 - BTS and Arbiter can communicate, but both cannot communicate with PTS/CTS
 - *Current triad voting first allows BTS to takeover as CTS and then Triad voting is disabled (design change)*
 - PTS and BTS can communicate, but both cannot communicate with Arbiter
 - *Triad voting is disabled (design change)*
 - PTS and Arbiter can communicate, but both cannot communicate with BTS
 - *Triad voting is disabled (design change)*
- Provides safeguard against:
 - Planned disruptive actions done sequentially on BTS and Arbiter
 - Double failures (Unplanned or combination of planned & unplanned) of BTS and Arbiter

Notes:

- This enhancement does not protect against double failures of both the PTS and BTS
 - Results in sysplex outage
- Best practice recommendation is for customer to reassign roles after failure affecting any of the special role servers

30


The STP recovery design has been changed to disable triad voting when the triad state is “degraded”. We still follow the rule that the PTS must surrender its role as the CTS when it cannot communicate with both the BTS and arbiter. This allows the BTS to take over as CTS when neither the BTS nor the arbiter can communicate with the PTS, as in the past. However, after the BTS takes over as CTS from a failed PTS, triad voting will be disabled. If triad voting were not disabled in this case, a subsequent failure of the arbiter could lead to a sysplex outage (assuming the CTN roles were not reassigned).

Triad voting is also disabled when:

- The PTS and BTS can communicate, but neither can communicate with the arbiter
- The PTS and arbiter can communicate, but neither can communicate with the BTS.

If triad voting is disabled, the BTS cannot take over as CTS using Arbiter Assisted recovery. Nor will the CTS surrender its role when it loses attachment to the remaining special role server. The BTS can still take over as CTS using either Console Assisted Recovery (CAR) or the STP Going Away Signal (GOSIG) transmitted from the CTS.

Triad voting is restored when the PTS, BTS, and arbiter can once again communicate with each other.

With this change, we get the advantages of the improved resiliency that is derived from triad voting, and we can fall back to traditional dyadic recovery mechanisms when such voting is not reasonable. Note that the enhancement to block disruptive changes to the BTS and the arbiter (in addition to the PTS) is important in this context as it helps prevent accidental disablement of the more desirable triad voting.

Note: It is still considered best practice to reassign the CTN roles prior to taking disruptive actions against relevant servers (the system enforces this) or after a failure of any such server. Doing so minimizes the potential for outages.

STP Recovery Enhancement



- Reliable unambiguous “going away” signal allows CTS in an STP-only CTN to notify BTS of its demise
- The BTS can then safely take over as the CTS
 - Dependencies on OLS and CAR removed in a 2 server CTN
 - Dependency on BTS > Arbiter communication removed in CTNs with 3 or more servers
 - BTS can also use GOSIG to take over as CTS for CTNs with 3 or more servers without communicating with Arbiter
- **Hardware Requirements**
 - STP-only CTN
 - z196 or z114
 - HCA3-O or HCA3-O LR connecting BTS and CTS

Driver 93 had an issue:
 Hiper EC.MCL N48165.053 (alert/circumvention)
 Hiper EC.MCL N48165.057 (alert/fix)
 False Going Away Signal detected in the BTS – fixed

31



Recovery enhancement: The new generation of host channel adapters (HCA3-O or HCA3-O LR), introduced for coupling, have been designed to send a reliable unambiguous "going away signal" (GOSIG) to indicate that the server on which the HCA3 is running is about to enter a failed (check stopped) state. When the GOSIG sent by the Current Time Server (CTS) in an STP-only Coordinated Timing Network (CTN) is received by the Backup Time Server (BTS), the BTS can safely take over as the CTS without relying on the previous recovery methods of Offline Signal (OLS) in a two-server CTN or the Arbiter in a CTN with three or more servers.


This enhancement is exclusive to the z196 and z114 and is available only if you have an HCA3-O or HCA3-O LR on the CTS communicating with an HCA3-O or HCA3-O LR on the BTS. Note that the already available STP recovery design is still used for cases when a GOSIG is not received, or for other failures besides a server failure.

In a Coordinated Timing Network (CTN) with only 2 servers/CFs, STP recovery is based on combination of a Server Offline Signal (OLS- Channel going away signal) and Console Assisted Recovery (CAR) . The Server Offline signal (OLS) is transmitted on a channel by the server to indicate that the channel is going offline. CAR is initiated when an OLS is not received. It uses the HMC/SE LAN path to determine if BTS can take over as CTS.


In a CTN with 3 or more servers/CFs, STP recovery employs an arbiter. If BTS loses communication on all established paths to CTS, the BTS and Arbiter communicate to establish if Arbiter also has lost communication on all established paths to CTS. If neither the BTS nor the arbiter can communicate with CTS, the BTS takes over as the CTS.

The critical point is that there can be at most one CTS in a CTN. The new support allows the BTS to reliably take over as CTS in more situations. This support is important for sysplex because it helps the system automatically overcome failures that might otherwise cause outages.

STP Resource Materials



- **Parallel Sysplex Web site**
 - www.ibm.com/systems/z/psa/stp.html
- **Redbooks®**
 - Server Time Protocol Planning Guide, SG24-7280
 - Server Time Protocol Implementation Guide, SG24-7281
 - Server Time Protocol Recovery Guide, SG24-7380
- **TechDocs**
 - Server Time Protocol Overview, PRS2398
 - STP Recovery White papers (search for STP)
- **Introduction to Server Time Protocol (STP)**
 - Course available on Resource Link
 - www.ibm.com/servers/resourcelink/hom03010.nsf/pages/education

32


The information at the Parallel Sysplex web site and the education course available on Resource Link provide high level overviews. The indicated Redbooks are a good source for details.

The **Redbooks** are available from links at the referenced Parallel Sysplex web site, or directly as follows:

Server Time Protocol Planning Guide

www.redbooks.ibm.com/redpieces/abstracts/sg247280.html

Server Time Protocol Implementation Guide

www.redbooks.ibm.com/redpieces/abstracts/sg247281.html

Server Time Protocol Recovery Guide

www.redbooks.ibm.com/redpieces/abstracts/sg247380.html

The **TechDoc** materials are available from links at the referenced Parallel Sysplex web site, or directly as follows:

Server Time Protocol Overview

www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS2398

STP Enhancement to Block Disruptive Actions

www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/WP102019

STP Enhancement - Recovery Voting

www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/WP102037

Important Considerations for STP server role assignments

www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/WP101833

The message based time synchronization protocol used by STP is similar to the Network Time Protocol (NTP), which is an industry standard. However, STP is not standard NTP. For further information on NTP and the NTP Public services project, refer to the Web sites: (1) www.ntp.org (2) support.ntp.org (3) www.faqs.org/rfcs/rfc1305.html

Agenda



- Hardware Updates
 - CFCC Level 17
 - InfiniBand (IFB) coupling links
 - Server Time Protocol (STP)
- Software Updates
 - z/OS V1R13 (new XCF Client/Server APIs)
 - z/OS V1R12
 - z/OS V1R11
- Summary

z/OS V1R13 - Summary



- **D XCF,SYSPLEX – Revised output**
- **CF Structure Placement – more explanation**
- **ARM – New timeout parameter for application cleanup**
- **XCF – New API for message passing**

- SETXCF MODIFY - Disable structure alter processing
- SDSF – Sysplex wide data gathering without MQ
- Runtime Diagnostics – Detects more contention
- zFS – Direct access to shared files throughout sysplex

Due to time restrictions, only the topic in bold will be discussed.
Slides for the remaining topics are included in the Appendix.

34



Summary of sysplex highlights for z/OS V1R13. Due to time constraints we will at best cover the topics in bold. Slides for the other topics are included in the appendix.

z/OS V1R13 - DISPLAY XCF,SYSPLEX



- D XCF,SYSPLEX command is a popular command used to display the systems in the sysplex
- But, prior to z/OS V1R13:
 - Output not as helpful for problem diagnosis as it could be
 - Much useful system and sysplex status information is kept by XCF, but not externalized in one central place
- So z/OS V1R13 enhances the output
 - You can still get the same output (perhaps with new msg #)
 - And you can get more details than before

35



D XCF produces the same output on all releases, a list of the names of the systems in the sysplex.

Many installations have set up automation to periodically issue D XCF,S. Prior to z/OS V1R13, this too produced the same list of names of systems in the sysplex as does D XCF. However, for diagnostic purposes, it would have been much more useful for the automation to issue D XCF,S,ALL instead. Prior to z/OS V1R13, D XCF,S,ALL listed the systems in the sysplex along with their status update TOD. When trying to diagnose sysplex problems, one often wants to know the state of the system. So in z/OS V1R13, the output of the D XCF,S command was changed to produce the equivalent of the D XCF,S,ALL command (instead of the simple list of system names). Thus for all those installations that periodically (or occasionally) issue D XCF,S to see the state of the sysplex, the system logs will now include the oh so vital status TOD of each system. Much more useful than the simple list of system names.

This change also prompted some reworking of the D XCF,S,ALL output in z/OS V1R13 to address some other serviceability and usability issues. The command now produces more detailed information about each system in the sysplex as well as information about the sysplex as a whole. The next slide has an example.

z/OS V1R13 – D XCF,SYSPLEX,ALL



z/OS 1.12	
D XCF,S,ALL	<pre>IXC335I 12:55:00 DISPLAY XCF FRAME LAST F E SYS=SY1 SYSPLEX PLEX1 SYSTEM TYPE SERIAL LPAR STATUS TIME SYSTEM STATUS SY1 4381 9F30 N/A 04/22/2011 12:55:00 ACTIVE TM=SIMETR SY2 4381 9F30 N/A 04/22/2011 12:54:56 ACTIVE TM=SIMETR SY3 4381 9F30 N/A 04/22/2011 12:54:56 ACTIVE TM=SIMETR SYSTEM STATUS DETECTION PARTITIONING PROTOCOL CONNECTION EXCEPTIONS: SYSPLEX COUPLE DATA SET NOT FORMATTED FOR THE SSD PROTOCOL</pre>
z/OS 1.13	
D XCF,S,ALL	<pre>IXC337I 12.29.36 DISPLAY XCF FRAME LAST F E SYS=SY1 SYSPLEX PLEX1 MODE: MULTISYSTEM-CAPABLE SYSTEM SY1 STATUS: ACTIVE TIMING: SIMETR NETID: 0F STATUS TIME: 05/04/2011 12:29:36.000218 JOIN TIME: 05/04/2011 10:31:08.072275 SYSTEM NUMBER: 01000001 SYSTEM IDENTIFIER: AC257038 01000001 SYSTEM TYPE: 4381 SERIAL: 9F30 LPAR: N/A NODE DESCRIPTOR: SIMDEV.IBM.PK.D13ID31 PARTITION: 00 CPCID: 00 RELEASE: z/OS 01.13.00 SYSTEM STATUS DETECTION PARTITIONING PROTOCOL CONNECTION EXCEPTIONS: SYSPLEX COUPLE DATA SET NOT FORMATTED FOR THE SSD PROTOCOL</pre>

36



This slide illustrates some of the new output in z/OS V1R13 for the DISPLAY XCF,SYSPLEX,ALL command, contrasting it with the output produced by the same command issued on a system running a prior release.

DISPLAY XCF will produce the same output on all releases (a list of system names). But on older releases, DISPLAY XCF,SYSPLEX and DISPLAY XCF both produced the same output since DISPLAY XCF,SYSPLEX was the default output for the DISPLAY XCF command.

In z/OS V1R13, issuing DISPLAY XCF,SYSPLEX will produce the same display output produced by issuing DISPLAY XCF,SYSPLEX,ALL on a prior release, except that the message number will be IXC336I instead of IXC335I.

Issuing DISPLAY XCF,SYSPLEX,ALL, or DISPLAY XCF,SYSPLEX,sysname on a z/OS V1R13 system will produce new output, which includes more detailed information about each of the specified systems. In addition, information about the sysplex as a whole will be produced.

z/OS V1R13 – CF Structure Placement



- Why did it put my structure in that CF ?
 - A dark art, often a mystery to the observer
- Existing messages updated to help explain
 - IXL015I: Initial/rebuild structure allocation
 - Also has “CONNECTIVITY=” insert
 - IXC347I: Reallocate/Reallocate test results
 - IXC574I: Reallocate processing, system managed rebuild processing, or duplexing feasibility

37



XES/XCF have added additional content to various messages in an attempt to clarify the reasons for making its CF Structure Allocation placement decisions.

XCF analyzes all available CFs from the preference list and ranks them according to weighted attributes they do or do not have. If two CFs have equal attributes they are ranked by their order in the preference list. The new inserts shown for each CF explains why that CF was positioned *below the previous one* in the eligibility queue.

z/OS V1R13 – CF Structure Placement



...

```

IXL015I STRUCTURE ALLOCATION INFORMATION FOR
STRUCTURE THRLST01, CONNECTOR NAME THRLST0101000001,
CONNECTIVITY=SYSPLEX
CFNAME      ALLOCATION STATUS/FAILURE REASON
-----
LF01        ALLOCATION NOT PERMITTED
            COUPLING FACILITY IS IN MAINTENANCE MODE
A           STRUCTURE ALLOCATED CC007B00
TESTCF      PREFERRED CF ALREADY SELECTED CC007B00
            PREFERRED CF HIGHER IN PREFLIST
LF02        PREFERRED CF ALREADY SELECTED CC007300
            EXCLLIST REQUIREMENT FULLY MET BY PREFERRED CF
SUPERSESES NO CONNECTIVITY 98007800
  
```

38



This slide shows some of the new inserts that could appear in one of the updated messages. This particular message is issued when the structure is allocated. It identifies the structure, the connector that caused it to be allocated, and the candidate coupling facilities. In this case, the structure was allocated in CF named A. The other coupling facilities were not selected for the indicated reasons.

PREFERRED CF HIGHER IN PREFLIST

This coupling facility is lower in the PREFLIST than another coupling facility that is suitable for allocation.

EXCLLIST REQUIREMENT FULLY MET BY PREFERRED CF

This coupling facility contains a structure from the EXCLLIST, and there is at least one coupling facility that does not contain any structures from the EXCLLIST.

Automatic Restart Management (ARM)



- If you have an active ARM policy, then:
 - After system failure, ARM waits up to two minutes for survivors to finish cleanup processing for the failed system
 - If cleanup does not complete within two minutes, ARM proceeds to restart the failed work anyway
- Problem: restart may fail if cleanup did not complete
- Issue: Two minutes may not be long enough for the applications to finish their cleanup processing

39



Automatic Restart Management (ARM) is a z/OS recovery function that can improve the availability of specific batch jobs or started tasks. When a job or task fails, or the system on which it is running fails, automatic restart management can restart the job or task without operator intervention. If a job or task fails, automatic restart management restarts it on the same system it was running on at the time of the failure. If a system fails, automatic restart management restarts the work on other systems in the sysplex; this is called a **cross-system restart**.

Before performing cross-system restart, ARM waits for member cleanup for the terminated system to complete. But, ARM will proceed with cross-system restart if cleanup takes longer than 2 minutes. Proceeding with cross-system restart before cleanup completes may cause cross-system restart of certain ARM elements to fail. For example, cross-system restarting DB2 using a command prefix will fail if the command prefix is still causing the command to be routed to the terminated system. In some environments and system termination scenarios, 2 minutes may not be long enough to complete member cleanup.

z/OS V1R13 – New ARM Parameter



- **CLEANUP_TIMEOUT**
 - New parameter for the ARM policy specifies how long ARM should wait for survivors to cleanup for a failed system
 - Specified in seconds, 120..86400 (2 min to 24 hours)
- **If parameter not specified**
 - Defaults to 300 seconds (5 minutes, not 2)
 - Code 120 if you want to preserve old behavior
- **If greater than 120:**
 - Issues message IXC815I after two minutes to indicate that restart is being delayed
 - If the timeout expires, issues message IXC815I to indicate restart processing is continuing despite incomplete cleanup
- **Available for z/OS V1R10 and up with APARs OA35357**

40



A new CLEANUP_TIMEOUT parameter is available for the ARM policy. The new CLEANUP_TIMEOUT parameter is used to determine how long ARM will wait for member cleanup for the terminated system to complete before performing cross-system restart.

With this support, by default, ARM will wait 5 minutes for member cleanup for a terminated system to complete before performing cross-system restart for an element. CLEANUP_TIMEOUT(120) must be added to the ARM policy in order to get the 2 minute timeout behavior that existed prior to this support. The new CLEANUP_TIMEOUT parameter can also be used to indicate that ARM can wait additional time for member cleanup for a terminated system to complete.

When a system terminates, automatic restart management updates the state of elements owned by the terminated system to FAILED. When member cleanup for the terminated system completes, systems targeted to perform cross-system restart update the state of eligible elements to RESTARTING and perform cross-system restart processing. When member cleanup for the terminated system does not complete within 2 minutes, systems that are targeted to perform cross-system restart update the state of eligible elements to RESTARTING and proceed to perform cross-system restart processing by using the CLEANUP_TIMEOUT parameter to introduce additional delay time, as necessary. The potential amount of additional delay time is any amount of time specified by the CLEANUP_TIMEOUT parameter that is more than 2 minutes.

Message IXC815I is issued to the system log to indicate cases where cleanup processing takes longer than two minutes (which implies restart processing is delayed). The message is also issued if the new cleanup timeout value expires. In that case, the message indicates that ARM is continuing with restart processing even though cleanup processing may not be complete. One possibility is that the timeout value needs to be increased to allow additional time for cleanup processing to complete. Another possibility is that the application has an error and failed to complete its cleanup, in which case problem diagnosis might be the reasonable way to proceed.

z/OS V1R13 – New XCF API for Message Passing



- XCF Client/Server Interfaces
 - Allows authorized programs to send and receive signals within a sysplex without joining an XCF Group
 - XCF does communication and failure handling
 - Simplifies development, reduces complexity, implementation and support costs by eliminating some of the XCF exploitation costs
 - Servers run in task mode

41



XCF provides a simplified set of interfaces for passing messages within a sysplex. These services allow a server to be established to process messages and for messages to be sent across the sysplex without first joining an XCF group. This is intended to make it easier to exploit XCF services for applications that do not require the member management and monitoring provided by the XCF group services interfaces.

z/OS V1R13 XCF Client/Server



- IXCSEND – send request to one or more servers
- IXCSRVR – start or stop a server instance
- IXCSEND – send response to client request
- IXCRECV – receive response(s) from server(s)

- IXCYSRVR – data mappings

42



IXCSEND, IXCSRVR, and IXCRECV are the executable macros that constitute the XCF Client/Server interface. The mapping macro IXCYSRVR defines the mappings for the various data areas.

User must be running supervisor state or with a PKM allowing key 0 to 7 (i.e., authorized), enabled, no locks held. Can be 31 or 64 bit mode, primary or AR mode.

IXCSEND and IXCRECV can be invoked in task mode or SRB mode, from any cross memory environment.

When invoking IXCSRVR to start a server instance, must be task mode, with P=H=S.

z/OS V1R13 Client/Server Basics



- Client must invoke IXCRECV on the same system that initiated the IXCSEND
 - But the send/rcv can be invoked from different work units running in different address spaces
- Servers are identified by name
 - Four 8 byte sections, case sensitive, start with component prefix
 - Externally presented in the form A.B.C.D
- Servers can have multiple instances
 - At various release levels
 - In various address spaces
 - On various systems
- All message data is delivered to user designated storage areas

43



The only real restriction on the client is that it invoke IXCSEND and IXCRECV from the same system. Unlike traditional XCF signalling services, where the msgout and msgin requests would be made from the member's address space, client/server applications have the freedom to do the send from one address space and the receive from another (if they so choose).

Servers are pretty much always identified by a "server name". For the programming interface, the name is a 32 byte string consisting of four 8 byte sections, where each section is left justified and padded on the right with EBCDIC blanks as needed. The first section is intended to be the component/vendor prefix so that different software applications will not have conflicting server names. This section must be non-blank. The remaining sections can be used as the application sees fit (subject to the rules on valid format). But one might for example expect the other sections to document the application and/or service somehow. It is up to the client/server application to figure out their own server naming convention so that clients can know what name to use when sending requests to a server.

With respect to XCF display output and input, the server names are always presented in "dot qualified" form. That is, we put a period between each of the sections and suppress the blanks.

A given server name can have more than one instance. An instance is essentially some task that claims to be the server. There can be more than one instance of the server on any system in the sysplex (that has this support). When sending a request, the client specifies the server name and the systems to which it should be sent. XCF on each of the target systems will select one of the server instances to process the request. We do not have any support that would allow a client to send a request to every instance of the server.

The client/server messages are always put into some user designated storage area. For traditional XCF signals, the message exit would be driven and the user would then invoke the msgin service to copy the data from an XCF buffer into user designated storage. When presenting a request to a server, XCF asks the server for the storage up front, XCF does the copy of the message into the user provided storage area, and then calls the server to process the message. When a client receives the results from the server, it tells XCF what storage area is to be used to receive those results.

z/OS V1R13 XCF Server



- A collection of tasks associated with a **server name**
- A server task:
 - Invokes **IXCSRVR** to define itself to XCF
 - Runs an infinite **XCF Server Stub** loop (until stopped)
 - XCF suspends the task if no work
 - XCF resumes task when work arrives
 - XCF calls **user provided server exit to process the work**
 - Is a **server instance**
 - Each instance has a unique **Server ID**
- A given server can have multiple instances to:
 - Handle volume of work
 - Support different application levels
 - Run in different spaces
 - Process selected request types
 - Process requests from certain clients or systems
- All definitions are maintained locally

44

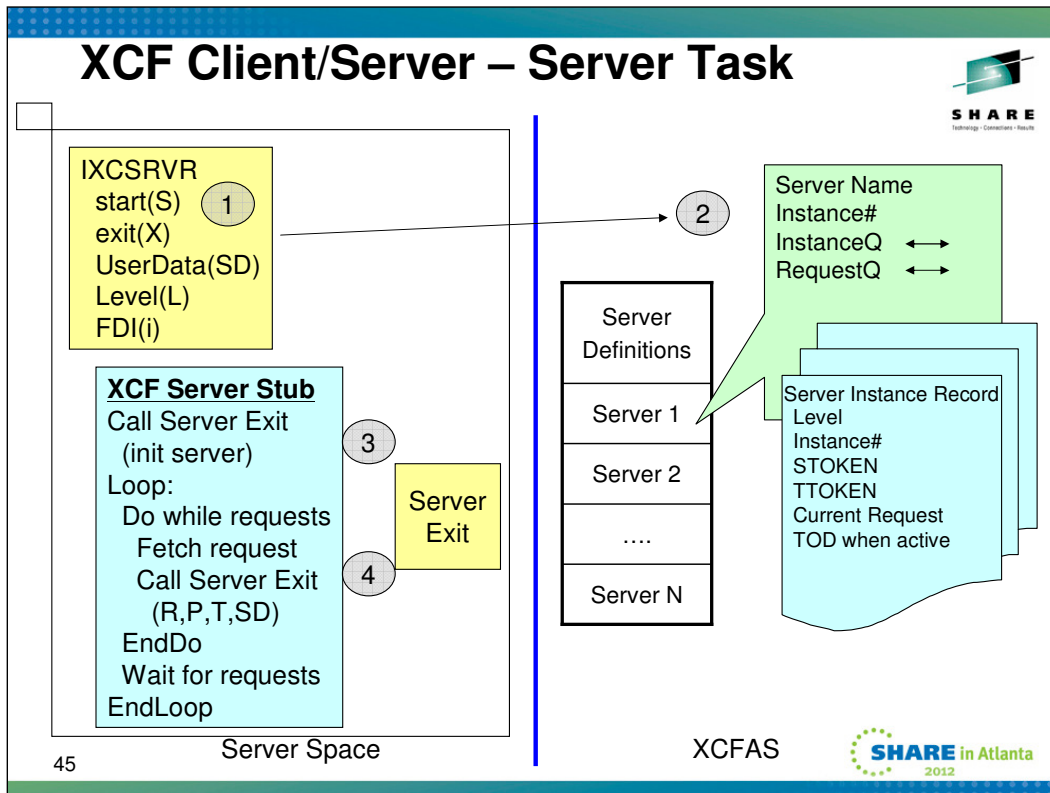


A "server" uses the IXCSRVR interface to define itself to XCF. The server must provide a "server exit" routine for XCF to call when there is a request for the server to process. If the server is successfully started, the service routine enters an infinite loop and will not return to the caller until the server is stopped.

One or more servers can be started for the same server name. Each such server is a server instance. The servers can be started in any address space on any system in the sysplex. Each server instance has a "server ID" that can be used to uniquely identify that instance (if need be, generally not)

A server instance is in essence a work unit running the "XCF server stub". This routine runs in an infinite loop, suspending when there are no requests to process. When a request arrives, the work unit is resumed and the XCF server stub calls the "server exit routine" to process the request.

All server definitions are local to the relevant system. We do not proactively propagate server definitions around the sysplex (unlike XCF group members). So for example, one would have to send a message to some other system to determine whether a particular server had been defined on that system.



- Under a task of the exploiter's choosing, the would be server invokes the IXCSRVR macro to start server S. The server indicates the address X of the server exit routine that XCF is to call to process a client request. The user data SD will be presented to the server exit routine. Typically exploiters use this data to locate a relevant control block. The level L indicates the level of functionality supported by the server. The intended use is to provide a way for a client (or XCF) to quickly determine whether the server has the functionality it requires. If the server is successfully started, the IXCSRVR macro will not return to the caller until the server is stopped. The server failure interval (FDI) indicates how long XCF should allow the server to appear unresponsive before it initiates action (kills task) to alleviate the apparent hang.
- The IXCSRVR service routine (not shown) PC's to the XCF address space to define the server. As needed, an entry is made in XCF's server definition table. A Server Instance Record is created to define the server instance. Multiple instances of the same server can be created. They need not be in the same address space. At most one instance of a particular server is permitted per task. Different server instances could be running at different levels. Multiple server instances might be required due to the volume of work. A server exit routine is not allowed to start a new server. That is, any one task can have but one server instance active at a time.
- After XCF successfully defines the server instance, control returns to XCF code running under the server task. The IXCSRVR service routine then calls the XCF Server Stub, who then makes the first call to the server exit routine for an "init server" request.
- The XCF Server Stub then starts the "endless" server loop. This stub routine will interact with XCF. It will fetch work items (client requests) queued to the server instance and call the server exit routine to process the work item. If the server is stopped, the XCF server stub will return to the IXCSRVR service routine, who will delete the relevant server instance and then return to the IXCSRVR invoker.

Key Concepts for Client Request



- Two ways to identify the target
 - Server name and system(s)
 - Server ID
- Receive Binds
 - Allow XCF to discard server responses if intended receiver terminates before invoking IXCRECV
 - Applies after traditional XCF sends client request to all targets
- Send Message Token
 - Identifies message, input to IXCRECV

46



Just in case we fail to make these points during prior discussion.

The client/server application needs to determine its server naming scheme and the mechanism by which clients are to discover (or know) the relevant server names. In most cases we expect clients to identify target servers by name and system. In such cases, there are a set of rules that determine how XCF goes about selecting an instance of the target server that is suitable for the request. Each server instance also has a unique Server ID. We anticipate that the Server ID might be used if a client and the server want to have a conversation such that once some server responds to the client request, the client might want to ensure that subsequent requests are sent to that specific server instance. The Server ID might also be used if the client/server application wants to override XCF's normal rules for selecting suitable servers. If the request is sent to a particular server instance, suitability of the server is the responsibility of the client.

A client that sends a request is expected to invoke IXCRECV to get the results of that request. However, the client could terminate before it gets a chance to do that. Since the client/server interfaces do not have any "membership", XCF does not have anything it can monitor to know that the client is gone. Thus we require each sender to specify a receive bind which in effect identifies some entity to be responsible for invoking IXCRECV. XCF will then notice when that entity terminates and do the appropriate cleanup. The main motivation is that we want to clean up XCF resources when the intended receiver terminates. Otherwise, the resources would persist until the message timed out (HOLDTIME). If the receiver happens to terminate before the traditional XCF signalling service has finished sending the request to all the targets, the message is allowed to persist until the sends are complete. This provides for a more consistent behavior (all the target servers will get the message).

The send message token returned by IXSEND is 32 bytes. The client needs to make the token available to whoever is expected to receive the server response as the token is needed when invoking IXCRECV to receive the server response.

Server Exit Processing



- Inputs (SXPL)
 - Base SXPL
 - Function Specific SXPL
 - SXPL_ServerCode
 - Initialize
 - Get Work Area
 - Process Request
- Outputs (via SXPL)
 - StopCode
 - State
 - WAD
 - RespBind
 - RefusalCode
 - ResultCode

48



The Server Exit Parameter List (SXPL) is passed to the server exit when called by XCF. It resides in storage obtained by XCF as private storage in the server address space. The SXPL is mapped by `ixcysrvr_tSXPL` which is defined in the `IXCYSRVR` macro. The SXPL contains a “base” portion that is common to all calls and a “function specific” portion that contains data unique to the particular function that the server exit is expected to perform. The `SXPL_ServerCode` contains the function code to indicate what function the server needs to perform. There are three possible functions:

Initialize – XCF calls the server exit to do whatever initialization or setup that may be needed. For many servers, this could be nothing. For others it might be reasonable to set up the work area that XCF will use when delivering client requests.

Get work area – XCF calls the server exit to have it provide a work area into which the text of the client request message will be stored. In the general case, the messages sent by clients could be of different sizes so one buffer (work area) may not be suitable for them all. Further, depending on the implementation of the server exit, it could be that each client request needs to be stored in its own buffer.

Process Request – After storing a copy of the client message in the user provided work area, XCF calls the server exit to process the request.

The server exit has certain SXPL fields that it is allowed to update. Not every field is allowed to be updated for every request. The `IXCYSRVR` macro documents the rules. The following fields can be updated:

Setting **SXPL_StopCode** causes XCF to stop the server (exit the server stub loop).

SXPL_State is 64 bytes of whatever the server exit wants to store. The content of this field is presented to the server exit each time it is called (initially zero). If the XCF server is used to query the server, this data is included in the results. This field is intended to document the “state” of the server in a way that is meaningful to the client/server application.

SXPL_WAD is the work area descriptor. It identifies the location and size of the work area to be used by XCF when it processes the next request. If no work area is provided, or if the work area is too small, XCF will call the server exit to get a work area for the request. Otherwise XCF will (re)use the work area. Thus server exits could have three models of operation: use the same work area over and over, have XCF ask for a new work area every time, or a mixture of both.

SXPL_RespBind can be updated to specify the response bind for the request. The response bind indicates what entity is responsible for sending the response. This field is likely of interest only to server exits that arrange for some other work unit to send the response to a request.

SXPL_RefusalCode is set to have XCF send an acknowledgment indicating that the server refused (rejected) the request.

SXPL_ResultCode is set to have XCF send an acknowledgment indicating the result of the server's processing. The server does not explicitly invoke `IXCSEND` to send a response if these codes are set.

Processing Client Request



- Servers likely to have one of the following behaviors
 - Server Exit invokes IXCSEND to send response
 - Server Exit arranges for 3d party to send response from:
 - Local system
 - Some other system
 - Server Exit sets SXPL ResultCode or RefusalCode to have XCF send an acknowledgment
 - Failure
- Two forms of XCF acknowledgment (zero, one, or both may apply)
 - Intermediate: request was delivered to server
 - Final: request is complete (response not expected)

49



This slide suggest several ways in which a server exit might deal with a client request. SDSF, for example, always deals with the request synchronously within the server exit.

If the server exit does not invoke IXCSEND to send its response, XCF will send an internal “intermediate ACK” back to the originating system to indicate that the request was in fact delivered to the server. Thus client/server applications can get a piece of information that traditional users of the XCF signalling service cannot. Namely, confirmation of delivery. This does not really do one any good in terms of knowing whether the request was processed successfully or not. To some extent the degree to which it is useful will depend on the design of the client/server application. At a minimum, it may help eliminate some of the possible failure scenarios that the client (or service personnel) might need to consider when a response is not received from the server.

In cases where the request is not processed by the server, or the server fails while processing the request, or the entity designated as responsible (via the “response bind”) for sending the response terminates, XCF will send a “final ACK” to the originator. A “final ACK” may cause the originating message to complete and allow the receiver to get the results. An intermediate ACK does not complete the message because we are still expecting a response from the server's agent.

Receiving Results



- Client invokes IXCRECV to get the results
- IXCRECV can be invoked to:
 - Test for message completion
 - Return results when message completes (blocking)
 - Report on progress so far (nonblocking)
- Outputs
 - Return and reason code
 - ANSAREA metadata to describe results
 - DATAAREA response data sent by servers

50

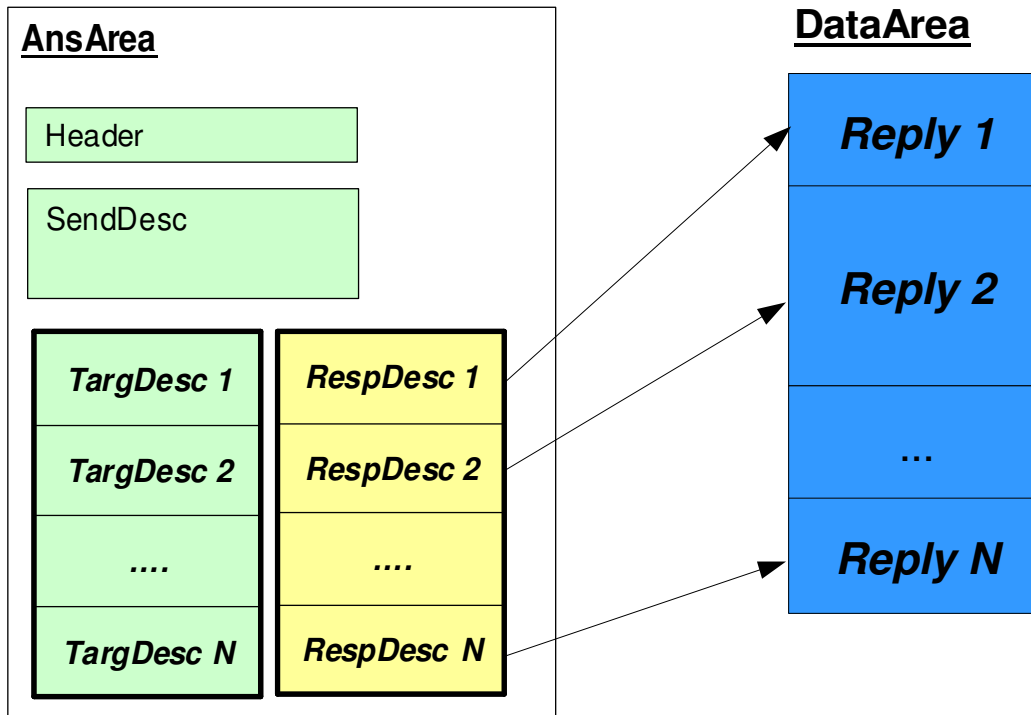


Clients can either poll for completion or perform a blocking receive wherein the caller's work unit will be suspended until the message is completed. If the message is already complete, the work unit will not be suspended. Only one work unit can be blocked for any particular request.

The client must receive the results before the HOLDTIME expires. HOLDTIME was specified when IXCSEND was invoked to send the request to the server. The timer for HOLDTIME starts as soon as the message is deemed to be complete (generally this means all expected responses have arrived, but it could also mean that RESPTIME has expired). If the HOLDTIME expires, the request and all its responses are subject to discard. If discarded, the IXCRECV service will indicate "no such message" if the client subsequently tries to receive the results.

The IXCRECV service has three primary outputs: a return and reason code to indicate the result of its processing, and as appropriate, an ANSAREA into which information about the results is stored, and a DATAAREA into which the text of the server reply messages is stored. Both ANSAREA and DATAAREA are storage areas provided by the receiver.

IXCRECV Answer Area



51



A successful IXCRECV RECEIVE=RESPONSES request will return an answer area in the storage area indicated by the ANSAREA keyword, and the text of any response messages in the storage area indicated by the DATAAREA keyword.

The answer area contains a header to indicate what, where, and how much data was returned by the IXCRECV service. If the answer area is too small to hold all the data, the header will indicate how much storage is needed. If there is enough space in the answer area, it will contain one send descriptor, one target descriptor for each target system, and if replies were expected, one response descriptor for each target system. The send descriptor contains data relevant to the IXCSEND request that was used to send the message. A target descriptor provides information about each potential target. A response descriptor provides information about each response and if (still) available, indicates where the text of that response can be found within the data area.

Response Codes



- Each response has a 2 byte response code that encapsulates XCF's understanding of the status of the message
- Primary response code indicates generic status
 - NotSent Request was never sent
 - InProgress Request is in flight
 - NoReceiver Server does not exist
 - NotDelivered Server never saw the request
 - Refused Server refused to accept the request
 - Failed Server failed while processing request
 - Delivered Request was presented to server
 - Replied Response received from server
- Secondary response code provides additional detail. For example, for "no receiver"
 - No instance of server is defined
 - No suitable instance of server is defined
 - Last suitable instance terminated

52



Not Sent: Message was not sent to target. Thus the target never processed the message and cannot possibly respond. The secondary response code explains why the message was not sent.

In Progress: The message was sent to the target. XCF does not have any further information about the message. It may or may not have been delivered to the target. If delivered to the target, the message may or may not have been processed successfully. If processed, the target may or may not have sent a response. If a response was in fact sent, it has not been received by the client system. This response code typically applies when the timeout values for the request expire.

No Receiver: There is no receiver for the message. The system to which the message was sent indicates that the intended target was not found.

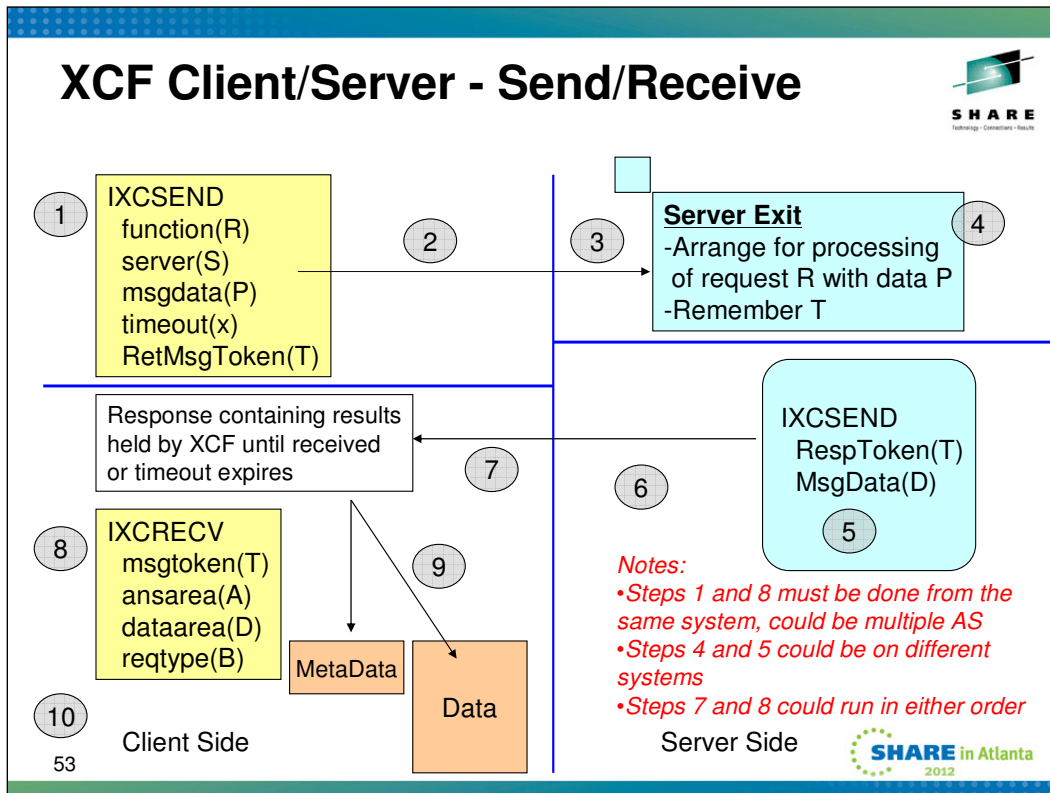
Not Delivered: Message was not delivered to the target. Suitable targets exist, but the message was neither presented to nor processed by any of them.

Refused: The target refused the message. The message was presented to the target server. The server set a nonzero "refusal code" to indicate that the message had been refused.

Failed: The target server failed while processing the message.

Delivered: The message was successfully delivered to the target. The message may or may not have been processed correctly by the target. If a response is expected, the response may or may not have been sent. If a response was sent, it has not been received.

Replied: Response received from target.



1. Client issues IXCSEND macro to send request R to server S, providing parameters P. The macro returns a token T that the client will later use to retrieve the results provided by the server. The parameters P can be at most 100 MB. The target server is identified by its name and the name of the system on which it resides. The same server name can be defined on any system in the sysplex.
2. XCF builds control blocks to manage the request and invokes IXCMMSGOX to send the request and its parameters to the system(s) on which the server (allegedly) resides.
3. XCF on the target system intercepts the signal. If the target server does not exist, the message is discarded and acknowledged with a "no receiver" response. If the server does exist, XCF invokes IXCMMSGC to save the message and queues a work item for the server. As needed, a server is selected and resumed. The server exit stub makes suitable preparations for processing the work item, including doing IXCMMSGIX to extract the client parameters from the saved message. XCF calls the server exit routine to present the request to the server.
4. The server exit routine inspects R to determine what type of request is to be processed. The server exit can process the request directly or it can arrange for it to be processed asynchronously. It needs to retain the token T that represents the client request for later use when sending the results of the request back to the originating client. Note that the token given to the client and the token given to the server represent the same logical request, but the tokens themselves will not have the same content.
5. After the server exit or its agent processes the request, the IXCSEND macro is invoked to send the results D back to the originating client. The token T identifies the client request to which the results belong. The results D can be at most 100 MB.
6. XCF decodes the token T to determine where the results should be sent. XCF invokes IXCMMSGOX to send the results to the originating system. If the message exceeds 60KB, XCF may suspend the responding thread until the IXCMMSGOX service finishes sending the message.
7. XCF on the client system intercepts the server response. XCF locates the control blocks used to manage the original client request. If not found, the client request timed out and the server response is discarded. Otherwise XCF binds the response message to the client request and holds it until the results are gathered by the client, or the client request times out. If the client is waiting for the results, XCF notifies (resumes) the client.
8. The client issues an IXCRECV request to gather the results of the request identified by token T (which was returned in step 1 when the request was initiated). The client provides an answer area A which will be filled with metadata that describes the response. The data area D will be filled with the response data sent by the server. It could be that the response has not yet arrived when the client attempts to gather it. The reqtype B indicates how the client wants to deal with this. The IXCRECV service can suspend (reqtype=blocking) until the response arrives (or the request times out or is cancelled), or the service routine can return immediately if the response is not yet available (reqtype=nonblocking).
9. To process the IXCRECV request, XCF locates the control blocks being used to manage the request identified by token T. If not found, the request timed out (or was cancelled) and no results are available. If the request is found, XCF determines whether the response has arrived or not. If not, XCF either suspends client thread or returns per the reqtype B. If suspended, XCF will resume the thread when the results arrive, or when the request times out, or when the request is cancelled, whichever comes first. Assuming the results have arrived, XCF determines whether the answer area and data area provided by the caller are large enough. If not, XCF returns to the client indicating what size is needed to receive the results. If the output areas are large enough, XCF fills them with the appropriate data, discards the control blocks used to manage the request, and returns to the client.
10. Client inspects Metadata and/or Data, and processes the results of the request.

DISPLAY XCF,SERVER



- The DISPLAY XCF command was extended to display information about servers, server instances, and queued work

```
D XCF, { SERVER | SRV }
    [ ,{SYSNAME | SYSNM}={sysname | (sysname [,sysname]. . .)} ]
    [ ,{SERVERNAME | SRVNAME | SRVNM}={servername} ]
    [ ,SCOPE={ {SUMMARY | SUM} | {DETAIL | DET} } ]
    [ ,TYPE=NAME [ , STATUS=(STALLED)] |
        {INSTANCE | INST}
        [ , STATUS=( [{WORKING | WORK}] [, STALLED] ) ]
        [ , {INSTNUM | INST#}=inst# ] ]
```

54



SERVER or SRV

Display information about servers in the sysplex. If you do not provide a qualifying operand, message IXC395I provides a list of all servers that are currently running.

If there are no active servers that match the filter criteria, the message IXC399I is shown.

SYSNAME or SYSNM=sysname(s)

Server information be displayed only for servers residing on the named systems.

SERVERNAME or SRVNAME or SRVNM=servername

Server information be displayed only for servers whose server names match the specified servername pattern. The servername may contain one or more wildcard character (*) and is case-sensitive when in quotes (' '). Periods can be used to separate server name sections. Blanks are assumed for sections not specified in the servername pattern. For example, *.* will match any server name that contains any characters in the first two sections and all blanks in the last two sections. Information for all servers is displayed if no servername is specified.

SCOPE=scope

Specifies the scope of server information be displayed:

SUMMARY or SUM

DETAIL or DET

TYPE=type

Specifies the type of server information be displayed:

NAME - Display information associated with servers that have the specific server name(s).

INSTANCE or **INST** - Display information associated with server instances.

STATUS=status

Requests that server information be displayed only for servers in the specified state:

STALLED

WORKING or **WORK** - This status can only be specified for TYPE=INSTANCE.

INSTNUM or INST#=inst#

Requests that server information be displayed only for server instances that have the specified instance number. This can only be specified for TYPE=INSTANCE.

Agenda




- Hardware Updates
 - CFCC Level 17
 - InfiniBand (IFB) coupling links
 - Server Time Protocol (STP)
- Software Updates
 - z/OS V1R13
 - z/OS V1R12
 - z/OS V1R11
- Summary

55



z/OS V1R12 Summary




- **REALLOCATE**
- Critical Members
- CFSTRHANGTIME
- Support for CFLEVEL 17
- Health Checks
- Auto Reply
- Run Time Diagnostics
- XCF Programming Interfaces

SFM History and Proven Practice
Wed 4:30 Session 10850

Due to time restrictions, only the topics in bold will be discussed.
Slides for the remaining topics are included in the Appendix

56



z/OS 1.12 functionality that may be of interest from a sysplex perspective.

Due to time restrictions, we may only be able to discuss the topics in bold. However slides for the remaining topics are included in the appendix for your consideration. Feel free to email the speaker if you have additional questions or comments.

Background - REALLOCATE



- **SETXCF START,REALLOCATE**
 - Puts structures where they belong
- **Well-received, widely exploited for CF structure management**
- **For example, to apply “pure” CF maintenance:**
 - SETXCF START,MAINTMODE,CFNAME=cfname
 - SETXCF START,REALLOCATE to move structures out of CF
 - Perform CF maintenance
 - SETXCF STOP,MAINTMODE,CFNAME=cfname
 - SETXCF START,REALLOCATE to restore structures to CF

57



The SETXCF REALLOCATE command is an existing system command used for CF structure management. The command causes the Coupling Facility Resource Manager (CFRM) component of XCF/XES to analyze existing coupling facility structures with respect to their placement in various CFs. The command then effects the necessary changes to correct any structure-related problems that it finds that are within its “scope of expertise.” As appropriate, the command may initiate actions (such as stop duplexing, rebuild, and/or start duplexing) in an orderly fashion to correct any problems it finds.

When a coupling facility is in maintenance mode, no new structures will be allocated in the CF and REALLOCATE will move allocated structures to an alternate CF per the preference list in the CFRM policy.

The SETXCF START,MAINTMODE command, which is available as of z/OS V1R9, in conjunction with the SETXCF START,REALLOCATE command can be used in combination to greatly simplify planned reconfiguration and maintenance actions. In particular, one need not update the CFRM policy in order to prevent structures from being allocated in the CF.

In the example illustrated in the slide, we have a case where (disruptive) CF maintenance is to be applied. The subject CF is placed in “maintenance mode” so that CFRM will refrain from allocating new structures in the CF. The REALLOCATE command takes maintenance mode into account, and concludes that structures need to be moved out of the subject CF. REALLOCATE will take the necessary actions to get the structures moved to an alternate CF (per the preference list).

Background - REALLOCATE



But...

- Difficult to tell what it did
 - Long-running process
 - Messages scattered all over syslog
 - Difficult to find and deal with any issues that arose
- And people want to know in advance what it will do

58



The existing SETXCF REALLOCATE command causes CFRM to analyze all allocated structures in the sysplex with respect to their placement in various CFs, and initiate the necessary changes to correct any structure-related problems that it finds. Each structure is successively evaluated and processed (as needed) one at a time so as to minimize disruption to the sysplex. As each structure is processed, REALLOCATE issues messages to describe its decisions and actions. In some cases, REALLOCATE is unable to successfully resolve a structure, and issues messages to so indicate. It is often the case that some form of manual intervention is needed in order to accomplish the desired reallocate of those structures. Prior to z/OS V1R12, the installation would have to search logs to find the messages for structures that had issues – a rather tedious task.

Furthermore, installations have also wanted to know in advance what actions reallocate was going to take. Depending on the structures to be manipulated, it might for example be desirable to delay the reallocate to a “slow” time of day so as to minimize the disruption to the exploiting application.

z/OS V1R12 - REALLOCATE



- DISPLAY XCF,REALLOCATE,option
- TEST option
 - Provides detailed information regarding what REALLOCATE would do if it were to be issued
 - Explains why an action, if any, would be taken
- REPORT option
 - Provides detailed information about what the most recent REALLOCATE command actually did do
 - Explains what happened, but not why

59



In z/OS V1.12 a new DISPLAY XCF,REALLOCATE,TEST command simulates the reallocation process and provides information about the changes that REALLOCATE (were it to be initiated by the SETXCF START,REALLOCATE command) would attempt to make, as well as any errors that it might encounter. This capability will provide information you can use to decide when to invoke the actual REALLOCATE process, and also whether you might need to make coupling facility configuration changes before issuing the actual REALLOCATE command.

For TEST, the CFRM policy is read into local storage (and it is not written back). The same processing that a real REALLOCATE would do is then applied against the in-store copy of the policy.

A new DISPLAY XCF,REALLOCATE,REPORT command will provide detailed information on the results experienced by a previously executed REALLOCATE command. This capability is intended to help you find such information without searching through the system log for REALLOCATE related processing and exception messages.

An actual REALLOCATE process stores “history” in the CFRM CDS for each structure defined in the policy. D XCF,REALLOC,REPORT reads the data and builds message text to reflect those results.

z/OS V1R12 – REALLOCATE ...



Caveats for TEST option

- Actual REALLOCATE could have different results
 - Environment could change
 - For structures processed via user-managed rebuild, the user could make “unexpected” changes
 - Capabilities of systems where REALLOCATE runs differ from the system where TEST ran
 - For example, connectivity to coupling facilities
- TEST cannot be done:
 - While a real REALLOCATE (or POPCF) is in progress
 - If there are no active allocated structures in the sysplex

60



The major things that could cause the TEST results to be different from the actual REALLOCATE:

- The environment could change between the TEST and the actual REALLOCATE. For example, the CFRM policy might be changed, or the set of connected coupling facilities might change.
- User-managed rebuild could do something "unexpected". When REALLOCATE initiates a user-managed rebuild, it has no real control over what the exploiter will actually do during the rebuild. For example, certain structure attributes and/or parameters could be changed by the user as part of the rebuild. Since TEST cannot know what the exploiter will really do during its processing, it makes a reasonable guess. If the guess is sufficiently divergent from what the exploiter actually does during the rebuild, the TEST results could differ from what the real REALLOCATE ultimately achieves.
- TEST and REALLOCATE might run on systems with different capabilities. TEST runs on exactly one system in the sysplex. REALLOCATE processing can run on different systems as it makes progress. Thus it could be that the system that processed the TEST has connectivity to a different set of CFs than one or more of the systems that participate in the actual REALLOCATE. Those differences could cause the REALLOCATE to put a structure in a different CF than the TEST expected.

TEST recognizes when it would not be worthwhile for it to run. If a REALLOCATE is currently in progress, the state of the structures is in flux and TEST cannot really make reliable predictions.

z/OS V1R12 – REALLOCATE ...



Caveats for REPORT option

- Can be done during or after a real REALLOCATE, but not before a real REALLOCATE is started
- A REPORT is internally initiated by XCF if a REALLOCATE completes with exceptions

61



Since the REPORT option gathers the data from the CFRM policy that describes the results of the most recent REALLOCATE, if the installation never performed a REALLOCATE, there would be nothing to report. Changes to the policy that result in structures being deleted will also delete the related history records. If a REPORT is requested while a real REALLOCATE is in progress, structures that have not yet been processed would appear in the “warning” section of the report.

If REALLOCATE completes with exceptions, XCF will internally initiate a DISPLAY XCF,REALLOCATE,REPORT request so that the results can be neatly summarized in the log. A comment on the DISPLAY command that is issued will indicate “ISSUED TO LOG REALLOCATE ERROR(S)/EXCEPTION(S)”.

Agenda



- Hardware Updates
 - CFCC Level 17
 - InfiniBand (IFB) coupling links
 - Server Time Protocol (STP)
- Software Updates
 - z/OS V1R13
 - z/OS V1R12
 - z/OS V1R11
- Summary

62



z/OS V1R11 - Summary



- **SFM with BCPii**
- System Default Action
- XCF FDI Consistency

SFM History and Proven Practice
Wed 4:30 Session 10850

Due to time restrictions, only the topic in bold will be discussed.
Slides for the remaining topics are included in the Appendix.

63



z/OS 1.11 functionality that may be of interest from a sysplex perspective.

Due to time restrictions, we may only be able to discuss the topics in bold. However slides for the remaining topics are included in the appendix for your consideration. Feel free to email the speaker if you have additional questions or comments.

System Default Action

In the absence of an SFM policy that explicitly specifies otherwise, assume ISOLATETIME(0) as the default action when a system becomes system status missing. Previously, the default action was PROMPT.

XCF FDI Consistency

Automatically adjust the Failure Detection Interval to be at least as high as that implied by excessive spin specifications

z/OS V1R11 - SFM with BCPii



- Expedient removal of unresponsive or failed systems is essential to high availability in sysplex
- XCF exploits new BCPii services to:
 - Detect failed systems
 - Reset systems
- Benefits:
 - Improved availability by reducing duration of sympathy sickness
 - Eliminate manual intervention in more cases
 - Potentially prevent human error that can cause data corruption

64

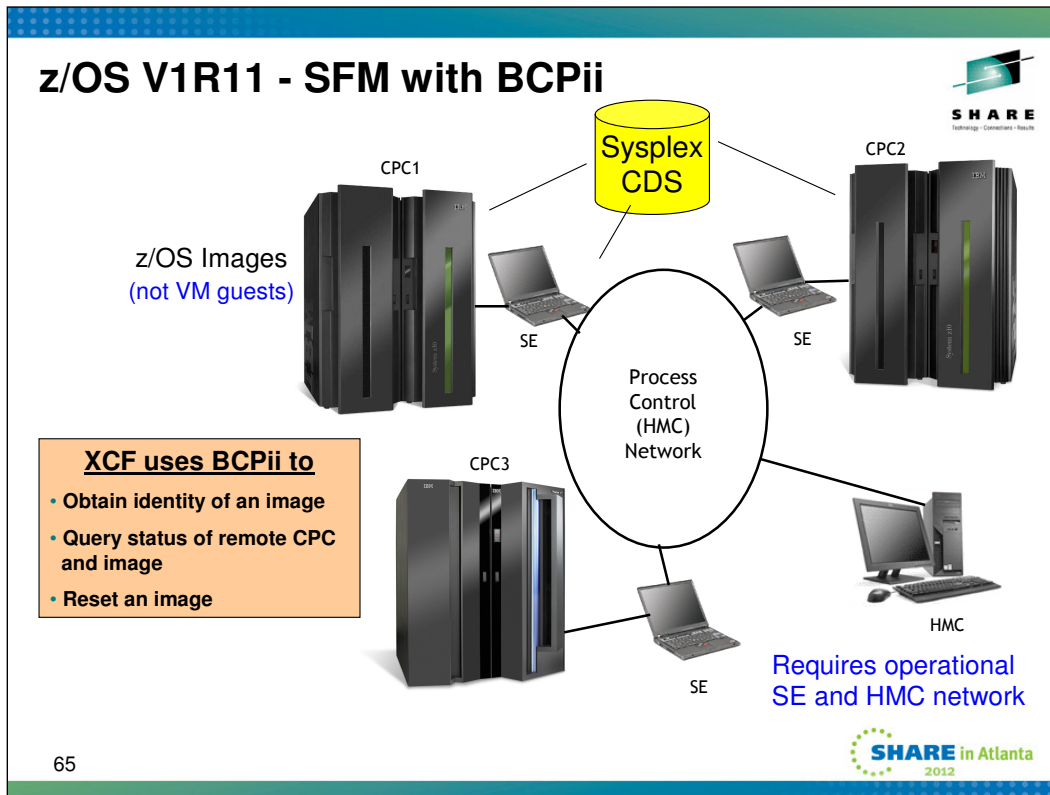


The sysplex failure management (SFM) component of XCF, which is used to manage failures and reconfiguration in the sysplex, has been enhanced in z/OS V1.11. It is now designed to use new Base Control Program internal interface (BCPii) services to determine whether an unresponsive system has failed, expedite sysplex recovery by bypassing delay intervals when possible, and automatically reset failed systems without manual intervention. This function allows SFM to avoid waiting for a period of time before assuming that systems have failed, improves the responsiveness of failure management, avoids operator intervention, and helps limit or avoid sysplex-wide slowdowns that can result from single-system failures.

The Base Control Program Internal Interface (BCPii) component of z/OS provides a set of programming interfaces to allow authorized programs to perform Hardware Management Console (HMC) functions for System z servers within an attached HMC network. These operations include obtaining information about servers and images (LPARs), issuing commands for certain hardware and software-related functions, and listening for certain hardware and software events. BCPii communication to HMCs and Support Elements (SEs) uses internal communication protocols and does not require communication on an IP network. Therefore, it is isolated from other network traffic.

Through the use of BCPii, XCF can detect that a system has entered a non-restartable wait-state, or that it has been re-IPLed. XCF can also perform a system reset on other systems in the sysplex. Thus XCF now has the ability to ascertain with certainty that a system is no longer operational. With this certain knowledge, XCF can ensure that the system is safely isolated from shared resources and remove the failed system from the sysplex – all without operator involvement. Furthermore, since XCF need not wait for the system failure detection interval to expire (to conclude that the system has no signs of life), the isolation of the failed system can occur sooner, which in turn reduces the amount of time that other systems in the sysplex will experience sympathy sickness.

BCPii is available on z/OS V1.10 with PTF UA47493, and requires a System z9 or z10 server with a microcode update. z114 and z196 servers are also supported. (For more information about required hardware levels, see the appropriate PSP bucket.) In z/OS V1.11, BCPii is designed to allow authorized programs to change or set data for certain HMC-managed objects associated with a CPC, Image, or Capacity Record. In addition, support for interactions with activation profile attributes were made available with the PTF for APAR OA29638.



CPC – Central Processor Complex containing images (LPARs)

SE – Support Element


HMC – Hardware Management Console

The Base Control Program internal interface (**BCPii**) allows authorized z/OS applications to have HMC-like control over systems in the process control HMC network. Note that there is complete communication isolation of existing networks (internet/intranet) from the process control (HMC) network, and communication with the System z support element is completely within base z/OS. BCPii provides a set of authorized APIs to enable communications between z/OS applications and the local support element, as well as between other support elements connected to other CPCs routed by the HMC. The BCPii query services can provide information about the operational state of any CPC connected to the HMC network, as well as the operational state of any image on the CPC.


As each z/OS image IPLs into the sysplex, XCF sets an IPL token in the hardware to uniquely identify the image. The IPL token is also published in the sysplex couple data set so that each system in the sysplex can ascertain the IPL token for every other system. If a system appears to be unresponsive, XCF uses BCPii query services to inquire as to the state of the subject system. If the system is down, it will be removed from the sysplex. As needed, XCF will use BCPii services to reset the system. For example, a system reset might be needed to ensure that the system has been successfully isolated from the sysplex. The IPL token is used when doing such resets, as it ensures that the reset is applied to the intended instance of the system image.


Note: If you hit the Emergency Power Off (EPO) switch on the CEC, everything on the CEC stops running. That includes the SE. If the SE is not running, it will not respond to BCPii queries and SFM will not be able to ascertain that the CEC is down. The sysplex reverts to old protocols.

z/OS V1R11 - SFM with BCPii



- With BCPii, XCF can know that system is dead, and:
 - Bypass the Failure Detection Interval (FDI)
 - Bypass the Indeterminate Status Interval (ISI)
 - Bypass the cleanup interval
 - Reset the system even if fencing fails
 - Avoid IXC102A, IXC402D and IXC409D manual intervention
 - Validate “down” to help avoid corruption of shared data



66


Unresponsive systems must be partitioned from the sysplex in a timely manner to avoid sympathy sickness. But, without BCPii, XCF does not really *know* a system’s operational state. At best, systems can monitor each other for signs of activity (updates to the sysplex couple data set, signals being exchanged). When the monitored activity stops, XCF waits the Failure Detection Interval (FDI) to try to avoid falsely removing an operational system. One would not want to remove a system that was suffering a temporary problem. But the penalty for this caution is that the sysplex may suffer workload processing interruptions/delays if the system has truly failed.


A partitioned system must be isolated from the rest of the sysplex to avoid corruption of shared data. In a parallel sysplex, XCF relies on CF Isolate command to fence a system from the channel subsystem. If the installation does not have a CF, or if the isolation fails, manual system reset is required. Manual intervention elongates the partitioning process, which elongates sympathy sickness.

But with BCPii services, XCF can now detect that a system has failed, and if so, whether it has been reset or otherwise isolated from shared resources. With this certain knowledge, XCF can safely remove a failed system from the sysplex without waiting for the failure detection interval to expire. If the failed system has not been isolated, XCF need not wait for the indeterminate isolation interval (ISI) to expire before it attempts to fence the failed system. If the fencing fails (or cannot be performed due to a lack of a CF), XCF can use BCPii services to appropriately reset the failed system. In cases where the failed system has been appropriately reset, XCF need not perform any isolation actions.

Thus the certain knowledge that a system has failed enables XCF to immediately partition failed systems from the sysplex, all without operator intervention.


ISI – can behave like ISOLATETIME(0) regardless of ISOLATETIME value specified in policy

z/OS V1R11 - SFM with BCPii



- SFM will automatically exploit BCPii as soon as the required configuration is established:
 - Pairs of systems running z/OS 1.11 or later
 - BCPii configured, installed, and available
 - XCF has security authorization to access BCPii defined FACILITY class resources or TRUSTED attribute
 - z10 GA2, or z196, or z114 (all with appropriate MCL's)
 - New version of sysplex CDS is primary in sysplex
 - Toleration APAR OA26037 for z/OS 1.9 and 1.10
 - Does NOT allow systems to use new SSD function or protocols

Enabling SFM to use BCPii will have a big impact on availability. Make it happen !

67


• See topic "Assigning the RACF TRUSTED attribute" in *MVS Initialization and Tuning Reference* for information on using RACF to assign the TRUSTED attribute to the XCF address space.

• Refer to the "BCPii Setup and Installation" topic in *MVS Programming: Callable Services for High Level Languages* for information on installation and configuration steps and SAF authorization requirements to enable BCPii to invoke z/Series Hardware APIs.

• A system running on z/OS V1R11 and down-level hardware is only eligible to target other systems that are enabled to exploit the full functionality of the SSD partitioning protocol. A system not running on the requisite hardware can not be the target of SSD partitioning protocol functions.

• Install toleration PTFs for OA26037 on V1R10 and V1R9 systems in the sysplex to use the newly formatted sysplex couple data set required by the protocol.

• By default, the SYSSTATDETECT function is enabled in V1R11. The current setting of the SYSSTATDETECT function can be determined by issuing a DISPLAY XCF,COUPLE command.

For more information on enabling or disabling the SYSSTATDETECT function in V1R11, see *MVS Initialization and Tuning Reference* for information on specifying SYSSTATDETECT in the COUPLExx parmlib member and *MVS System Commands* for information on enabling and disabling the SYSSTATDETECT function via the SETXCF FUNCTIONS command

SSD = SYSSTATDETECT = System Status Detection

Agenda



- Hardware Updates
 - CFCC Level 17
 - InfiniBand (IFB) coupling links
 - Server Time Protocol (STP)
- Software Updates
 - z/OS V1R13
 - z/OS V1R12
 - z/OS V1R11
- **Summary**

Highlights



- CFLEVEL 17 for z196 and z114
 - More structures and nondisruptive dumping
- Infiniband links for
 - Bandwidth
 - High performance links at 150 meters
 - Fewer physical links
 - Additional connectivity with HCA3-O LR (four ports)
- STP recovery enhancements
- Automatic resolution of sympathy sickness
 - **SFM with BCPii for better availability**
 - CFSTRHANGTIME, Critical Members
- CF Structure Management
 - REALLOCATE test and report

69



Coupling Facility Control Code (CFCC) CFLEVEL 17 for the z196 and z114 supports up to 2047 structures. Nondisruptive dumping enables capture of diagnostic data without disruption.

Infiniband links can be used for increased bandwidth, can help simplify sysplex configurations by reducing the number of links needed to connect CECs, and can provide high performance links at greater distances than do current links.

STP recovery enhancements help make the STP-only CTN more resilient.

SFM with BCPii is a critical technology for improving sysplex availability as it allows XCF to know with certainty that an apparently unresponsive system is in fact not operational. This knowledge enables XCF to remove systems from the sysplex without operator intervention.

z/OS V1R12 extends Sysplex Failure Manager (SFM) support to provide automatic resolution of additional sympathy sickness conditions which would otherwise impact the sysplex. It also provides some enhancements related to the Coupling Facility Resource Manager (CFRM) REALLOCATE function that customers have requested, namely the ability to determine what the function might do and what it most recently did.

Other Sysplex Related Sessions



- Tue 4:30 **Introducing z196 and z114 PCIe I/O and coupling infrastructure**

- Wed 1:30 **Coupling Technology Overview and Planning**
- Wed 3:00 **z/OS Planned Outage Avoidance Checklist**
- Wed 4:30 **SFM History and Proven Practice**
- Wed 4:30 **Migrating from ICB4 to Infiniband**

- Thu 8:00 **Migrating from z10 ICBs to z196 Infiniband**
- Thu 9:30 **z/OS 1.13 SDSF Update**

70



Other sessions of note this week. Most of these sessions provide additional detail on topics mentioned briefly in this presentation.

z/OS Publications



- *MVS Setting Up a Sysplex (SA22-7625)*
- *MVS Initialization and Tuning (SA22-7591)*
- *MVS Systems Commands (SA22-7627)*
- *MVS Diagnosis: Tools and Service Aids (GA22-7589)*
- *z/OS V1R13.0 Migration (GA22-7499)*
- *z/OS V1R13.0 Planning for Installation (GA22-7504)*
- *z/OS MVS Programming: Callable Services for High Level Languages (SA22-7613)*
 - Documents BCPii Setup and Installation and BCPii APIs
- *Migration to the IBM zEnterprise System for z/OS V1R7 through z/OS V1R12 (SA23-2269)*

71



These publications are available at <http://www.ibm.com/systems/z/os/zos/bkserv/>

Sysplex-related Redbooks



- **System z Parallel Sysplex Best Practices, SG24-7817**
- **Considerations for Multi-Site Sysplex Data Sharing, SG24-7263**
- **Server Time Protocol Planning Guide, SG24-7280**
- **Server Time Protocol Implementation Guide, SG24-7281**
- **Server Time Protocol Recovery Guide, SG24-7380**
- **System z Parallel Sysplex Performance, SG24-7654**

- **Exploiting the IBM Health Checker for z/OS Infrastructure, REDP-4590**

- **Available at www.redbooks.ibm.com**

72



Redbooks often provide clear, concise, comprehensive material.

Parallel Sysplex Web Site



<http://www.ibm.com/systems/z/advantages/psa/index.html>

Parallel Sysplex

IBM SERVER TIME PROTOCOL (STP)
Time Synchronization for the Next Generation
[→ Learn more](#)



About | **STP** | **Supporting products** | **Learn more** | **Services**

Overview | Detailed info | Benefits | What's new |
CF structures | CF levels | IFB



Appendix

Material of potential interest, even though it was not presented due to time constraints.

Appendix – z/OS V1R13



- SETXCF MODIFY - Disable structure alter processing
- SDSF – Sysplex wide data gathering without MQ
- Runtime Diagnostics – Detects more contention
- zFS – Direct access to shared files throughout sysplex

These topics were omitted from the presentation due to time restrictions.

CF Structure Alter Processing



- CF Structure Alter processing is used to dynamically reconfigure storage in the CF and its structures to meet the needs of the exploiting applications
 - Size of structures can be changed
 - Objects within structures can be reapportioned
- Alter processing can be initiated by the system, the application, or the operator
- There have been occasional instances, either due to extreme duress or error, where alter processing has contributed to performance problems
- Want an easy way to inhibit alter processing

76



Alter processing allows the sysplex to dynamically react to the needs of the workload. For example, the size of a CF structure can be increased automatically when it starts to get full. Doing so could, for example, help the sysplex accommodate a spike in work. CF Structures contain several different types of objects and different workloads might require varying numbers of these objects to run optimally. Alter processing allows the objects in the structure to be tuned for optimal processing.

The ALLOWAUTOALT specification for each structure in the active Coupling Facility Resource Manager (CFRM) policy determines whether the system will automatically initiate alter processing for any give structure. XES offers the IXLALTER programming interface which allows an application to initiate alter processing as it sees fit. Finally, the installation can manually initiate alter processing through use of the SETXCF START,ALTER command.

There have been, on occasion, instances where it would have been desirable to prevent alter processing from starting. In such cases, one could certainly refrain from issuing the SETXCF command to initiate alter processing. One could also format and activate a new Coupling Facility Resource Management (CFRM) policy to not allow the system to automatically start alter processing. However, one had no ability to prevent applications from initiating alter processing via the programming interface. And updating the CFRM policy is probably not really something one would want to do anyway. So it would be nice to have a simple technique for disabling alter processing, and which provided complete coverage.

z/OS V1R13 – Enable/Disable Start Alter Processing



- SETXCF MODIFY,STRNAME=pattern,ALTER=DISABLED
- SETXCF MODIFY,STRNAME=pattern,ALTER=ENABLED
 - STRNAME=strname
 - STRNAME=strprfx*
 - STRNAME=ALL | STRNAME=*
- D XCF,STRUCTURE, ALTER={ENABLED|DISABLED}
- Only systems with support will honor ALTER=DISABLED indicator in the active policy
 - So you may not get the desired behavior until the function is rolled around the sysplex
 - But fall back is trivial since downlevel code ignores it
- APAR OA34579 for z/OS V1R10 and up
 - OA37566 as well

77



The SETXCF MODIFY,STRNAME=xxx,ALTER=DISABLED command can be used to update the active instance of the current CFRM policy to indicate that alter processing is not to be started for the named structures. This indicator does not change your CFRM policy definition, it is a flag in the “operational” copy of the policy that is used by the system. Note that this command will prevent a new alter from being started. It will not impact an ongoing alter request. One can use SETXCF STOP,ALTER,STRNAME=strname to stop ongoing alter processing for a structure.

Use SETXCF MODIFY,STRNAME=xxx,ALTER=ENABLED to update the indicator in the active CFRM policy so that alter processing for the named structures is once again allowed to start.

The DISPLAY XCF,STRUCTURE output includes a line to indicate that requests to start a new alter will not be processed for the subject structure. The absence of the line denotes the fact that a new alter can be started. A new filter, ALTER=ENABLED or ALTER=DISABLED, will cause the DISPLAY XCF,STRUCTURE command to include only those structures for which start alter processing is enabled or disabled, respectively.

```
SETXCF MODIFY,STRNAME=*,ALTER=DISABLED
IXC556I SETXCF COMMAND COMPLETED: ALTER DISABLED FOR 128 STRUCTURE(S).
D XCF,STR,ALTER=DISABLED
IXC359I 16.43.44 DISPLAY XCF
STRNAME          ALLOCATION TIME      STATUS                TYPE
BIGONE           --          --          NOT ALLOCATED
                START ALTER NOT PERMITTED
CACHE01          05/12/2011 16:42:34 ALLOCATED             CACHE
                START ALTER NOT PERMITTED
```

The “do not start new alters” indicator will persist across sysplex IPLs.

NOTE: Alter processing is an important tool for autonomic tuning within the sysplex. Disabling it could have negative impacts on the sysplex. So in general, one would not want to leave it disabled for long periods of time.

With APAR OA34579, the system fails to automatically contract structures when the CF is more than 90% full. Install OA37566 to fix.

z/OS V1R13 - SDSF



- SDSF provides sysplex view of panels:
 - Health checks; processes; enclaves; JES2 resources
- Data gathered on each system using the SDSF server
- Consolidated on client for display so user can see data from all systems
- Previously used MQ series to send and receive requests
 - Requires configuration and TCP/IP, instance of MQ queue manager on each system
- z/OS V1R13 implementation uses XCF Client/Server
 - No additional configuration requirements

78



In z/OS V1R13, SDSF adds new support and removes the requirement for IBM WebSphere® MQ for z/OS (5655-L82) in JES2 environments once all systems in a MAS are running z/OS V1.13 JES2. In this release, SDSF is designed to implement for JES3 all applicable functions that are supported for JES2. For JES2, new planned support includes JES network server and network connections displays. Once all systems in a JES3 complex are using z/OS V1.13 JES3, the new support includes displays for initiators, output, held information, job 0, punches, readers, JES network server, and JES Network Connections. The corresponding SDSF Java classes are updated to support the new displays and actions. These changes are intended to provide systems management improvements.

If want a sysplex wide view in a mixed sysplex, need to continue to use MQSeries for all systems. If willing to limit “sysplex-wide” to just those systems running z/OS V1R13, one need not use MQSeries. Otherwise not until all systems are running with z/OS V1R13 can the requirement for MQSeries be eliminated.

z/OS V1R13 – Runtime Diagnostics



- Allows installation to quickly analyze a system experiencing “sick but not dead” symptoms
- Looks for evidence of “soft failures”
- Reduces the skill level needed when examining z/OS for “unknown” problems where the system seems “sick”
- Provides timely, comprehensive analysis at a critical time period with suggestions on how to proceed

- Runs as a started task in z/OS V1R12
 - S HZR
- Starts at IPL in z/OS V1R13
 - F HZR,ANALYZE command initiates report

79



z/OS Runtime Diagnostics (RTD) was introduced in z/OS V1R12 to help you when the need for quick decision-making is required. Runtime Diagnostics analyzes key system indicators of a running system. The goal is to help you identify the root of problems that cause system degradation on systems that are still responsive to operator commands. Runtime Diagnostics runs quickly enough to return results fast enough to aid you in making decisions about alternative corrective actions and facilitate high levels of system and application availability.

Run Time Diagnostics will identify critical messages, search for serialization contention, find address spaces consuming a high amount of processor time, and analyze for patterns common to looping address spaces.

When introduced in z/OS V1R12, one issued a START command to have RTD perform its analysis. But in some situations, a system might be so sick that it would not be able to start RTD. Such issues are mitigated in z/OS V1R13 by having RTD start at IPL. In z/OS V1R13, you issue a MODIFY command (F HZR,ANALYZE) to have RTD perform its analysis. The system still has to be healthy enough to process commands, but that is generally less taxing than what would be needed to initiate a started task.

Runtime Diagnostics may prove quite useful when XCF surfaces various hang conditions or situations for which sympathy sickness might arise. That is, those situations that are to be addressed automatically via SFM parameters such as MEMSTALLTIME and CFSTRHANGTIME.

For more information, see z/OS V1R13 Problem Management (G325-2564)

z/OS V1R13 – Runtime Diagnostics ...



Does what you might do manually today:

- Review critical messages in the log
- Analyze contention
 - GRS ENQ
 - [GRS Latches](#)
 - [z/OS UNIX file system latches](#)
- Examine address spaces with high CPU usage
- Look for an address space that might be in a loop
- Evaluate local lock conditions
- Perform additional analysis based on what is found
 - For example, if XES reports a connector as unresponsive, RTD will investigate the appropriate address space

80



Critical Message Analysis: Reads through the last hour of OPERLOG looking for critical messages. If any are found, lists the critical message as an error event. For a subset of critical messages, performs additional analysis based on the message identifier and the content of the message.

Contention Analysis:

- Provides a point in time check of **GRS ENQ** contention equivalent to issuing the D GRS,AN,WAITER command. Compares the list of job names that are waiters with the list of system address spaces that are started at IPL to determine if any system address spaces are waiters. If ENQ contention is found, issues an error event message.
- Reports when **GRS Latches** are in contention.
- Reports when **z/OS UNIX latch contention or waiting threads** exist for more than five minutes.

CPU analysis: Provides a point in time check of any address space that is using more than 95% of the capacity of a single CPU, which might indicate the address space is in a loop.

Loop Detection: Looks through all tasks in all address spaces to determine if a task appears to be looping. Examines various system information for indicators of consistent repetitive activity that typically appears when a task is in a loop. When both a HIGHCPU event and a LOOP event list the job name, there is a high probability that a task in the job is in a loop.

Local Lock Contention: Provides a point in time check of local lock suspension for any address space. Calculates the amount of time an address space is suspended waiting for the local lock. If an address is suspended more than 50% of the time waiting for a local lock, issues an event message.

z/OS V1R13 - zFS



- Full read/write capability from anywhere in the sysplex for shared file systems
 - Better performance for systems that are not zFS owner
 - Reduced overhead on the owner system
- Expected to improve performance of applications that use zFS services
 - z/OS UNIX System Services
 - WebSphere® Application Server

81



zFS processing has been redesigned to allow all members of a Parallel Sysplex to perform zFS file system read and write I/O operations for shared file systems. This is expected to yield substantial performance gains for systems that would not have been zFS owning systems in the prior design, without performance impacts to systems that would have been zFS owning systems.

Applications that use zFS, such as z/OS UNIX System Services and WebSphere Application Server for z/OS, are expected to benefit.

Applies when all systems are running z/OS V1R13. In such cases, zFS can directly read and write user data to a sysplex-aware (RWSHARE) read-write file system. This generally improves the performance of client system access to the files since the data does not need to be sent between systems via XCF communications. Only metadata updates are sent to the zFS owning system.

Appendix – z/OS V1R12



- Critical Members
- CFSTRHANGTIME
- Support for CFLEVEL 17
- Health Checks
- Auto Reply
- XCF Programming Interfaces

82



These topics were omitted from the presentation due to time restrictions.

Sysplex Failure Manager (SFM) takes action to alleviate sympathy sickness if a “critical” member of an XCF group is demised.

The CFSTRHANGTIME specification enables SFM to take action to alleviate sympathy sickness when a connector to a CF structure is unresponsive.

The z/OS support for CFLEVEL 17 was touched upon in the main presentation. Here we provide additional details.

We describe the new XCF/XES health checks introduced in z/OS V1R12.

We talk about the Auto Reply support which can provide some simple message automation during IPL.

We mention new XCF programming interfaces introduced in z/OS V1R12.

z/OS V1R12 - Critical Members



- A system may appear to be healthy with respect to XCF system status monitoring, namely:
 - Updating status in the sysplex CDS
 - Sending signals
- But is the system actually performing useful work?
 - There may be critical functions that are non-operational
 - Which in effect makes the system unusable, and perhaps induces sympathy sickness elsewhere in the sysplex
- Action should be taken to restore the system to normal operation OR it should be removed to avoid sympathy sickness

83



z/OS V1R12 extends XCF System Status monitoring to incorporate status information about critical components (such as GRS). Currently, a system is deemed unresponsive if it stops sending XCF signals and stops updating its status in the sysplex Couple Data Set (CDS). However, these indications of activity do not necessarily imply that a system is able to accomplish useful work. Indeed, an apparently active system could in effect be causing sympathy sickness because critical components are unable to accomplish their intended function. The goal of the “critical member” support is to resolve the sympathy sickness by expeditiously partitioning a sick system out of the sysplex whenever any critical XCF member on that system is deemed unresponsive. Though still not a perfect indicator of whether a system is performing useful work, the discovery of unresponsive critical components should provide an incremental improvement that helps the sysplex better identify (and remove) unresponsive systems.

z/OS V1R12 - Critical Members ...



- **A Critical Member is a member of an XCF group that Identifies itself as “critical” when joining its group**
- **If a critical member is “impaired” for long enough, XCF will eventually terminate the member**
 - Per the member’s specification: task, space, or system
 - SFM parameter MEMSTALLTIME determines “long enough”
- **GRS is a “system critical member”**
 - XCF will remove a system from the sysplex if GRS on that system becomes “impaired”

84



z/OS V1R12 extends XCF Member Status monitoring to take some form of action when a critical XCF group member appears to be non-operational. XCF externalizes via messages that the member is “impaired”. Furthermore, XCF will terminate the critical member if the impaired state persists long enough.

Surfacing the condition should make it easier to identify situations where an application may not be operating normally. Termination of the critical member should relieve the sympathy sickness condition and allow the application to resume normal operation. Alternatively, such termination may also make it possible for more timely restart of the application (or other appropriate recovery action) that can then lead to full recovery. There is some danger that such termination could negatively impact the application, however, that is something for the application writer to assess and exploit as appropriate. The application determines whether it is “critical” and if so, the means by which it should be terminated. Said termination could entail termination of the member’s task, address space, or system. If the system is to be terminated, the member is presumed to be “system critical”. GRS is “system critical”.

If a critical member remains continuously impaired for as long as the system failure detection interval (FDI), XCF inspects the Sysplex Failure Manager (SFM) specification for the MEMSTALLTIME parameter. For MEMSTALLTIME(NO), XCF delays termination of the member for FDI seconds, or two minutes, whichever is longer. If MEMSTALLTIME(nnn) is specified, XCF delays termination for the indicated number of seconds).

This function is intended to help reduce the incidence of sysplex-wide problems that can result from unresponsive critical components. GRS exploits these XCF critical member functions in both ring and star modes. GRS monitors its key tasks and notifies XCF if it detects that GRS is impaired.

z/OS V1R12 - Critical Members ...



- New Messages
 - IXC633I “member is impaired”
 - IXC634I “member no longer impaired”
 - **IXC635E “system has impaired members”**
 - IXC636I “impaired member impacting function”
- Changed Messages
 - IXC431I “member stalled” (includes status exit)
 - IXC640E “going to take action”
 - IXC615I “terminating to relieve impairment”
 - IXC333I “display member details”
 - IXC101I, IXC105I, IXC220W “system partitioned”

85



This slide summarizes the essence of the new and changed messages related to impaired member processing.

Installations may choose to develop automation and/or operational procedures to deal with impaired member conditions. The Sysplex Failure Management (SFM) policy MEMSTALLTIME specification should be specified accordingly. For example, if an installation wants operators to investigate and resolve such problems, one will likely specify a longer MEMSTALLTIME value to allow time for such actions to occur. MEMSTALLTIME in effect serves as a back stop to allow the system to take automatic action to alleviate the problem if the operations personnel are unable to resolve the problem in a timely manner. Of course one should recognize that the higher the MEMSTALLTIME value the longer the potential sympathy sickness impact on other systems in the sysplex will persist.

Message IXC635E is likely the key message that would be used to trigger the relevant automation and/or operational procedures.

The sysplex partitioning messages (IXC101I “system being removed”, IXC105I “system has been removed”, and IXC220W “XCF wait-stated system”) have new inserts to indicate that the system was removed as the result of terminating an impaired member. The XCF wait-state code 0A2 has a new reason code (x194) to indicate this condition as well.



z/OS V1R12 - Critical Members ...

- **Coexistence considerations**

- Toleration APAR OA31619 for systems running z/OS V1R10 and z/OS V1R11 should be installed before IPLing z/OS V1R12
- The APAR allows the down level systems to understand the new sysplex partitioning reason that is used when z/OS V1R12 system removes itself from the sysplex because a system critical component was impaired
- If the APAR is not installed, the content of the IXC101I and IXC105I messages will be incorrect

86



The z/OS V1R12 system has a new sysplex partitioning reason code and a new 0A2 wait-state reason when a system is removed from the sysplex. We now have so many reasons for killing systems that we have exhausted the internal data structures that were used to share this information among systems in the sysplex. To add the new “impaired member” reason, we had to make changes that will cause down-level systems to issue partitioning messages with completely misleading inserts. For example, a down level system would issue the following messages if a z/OS V1R12 system was removed from the sysplex to terminate an impaired critical member:

```
IXC101I SYSPLEX PARTITIONING IN PROGRESS FOR SY4 REQUESTED BY
XCFAS. REASON: LOSS OF COUPLE DATA SET
```

```
IXC105I SYSPLEX PARTITIONING HAS COMPLETED FOR SY4
- PRIMARY REASON: LOSS OF COUPLE DATA SET
- REASON FLAGS: 800015
```

The toleration APAR allows the down level systems to issue messages with the correct text. For example:

```
IXC101I SYSPLEX PARTITIONING IN PROGRESS FOR SY4 REQUESTED BY
XCFAS. REASON: SYSTEM HAS AN IMPAIRED CRITICAL MEMBER
```

We believe the only detrimental impact of running without the toleration APAR is the misleading messages. Still, we recommend that it be installed before a z/OS V1R12 system is IPL'ed into the sysplex.



z/OS V1R12 - Critical Members ...

- **Potential migration action**
 - Evaluate, perhaps change MEMSTALLTIME parameter

87



If you do not currently have an active SFM policy, or your SFM policy does not specify MEMSTALLTIME, MEMSTALLTIME(NO) is the default. With MEMSTALLTIME(NO), SFM will terminate an impaired critical member after the system failure detection interval (FDI) or two minutes, whichever is greater. The default FDI on a z/OS V1 R12 system is likely 165 seconds. You can issue the DISPLAY XCF,COUPLE command to see the FDI (aka INTERVAL) value that is being used by your system.

If your active SFM policy specifies MEMSTALLTIME(n) where “n” is some integral number of seconds, that value “n” will determine the number of seconds that SFM waits before it terminates an impaired critical member that is deemed to be impacting its function (and thus the system, and thus the sysplex). This specification is likely suitable for z/OS V1R12 as well. The rationale used to pick a MEMSTALLTIME value for dealing with signalling sympathy sickness conditions is likely valid for critical member support as well. Namely, a situation has occurred in which the system has recognized that there might be a sympathy sickness impact. MEMSTALLTIME indicates how long XCF should delay before taking action to alleviate the condition (i.e. terminate the member). However much time was needed for automation and/or operational procedures to run their course for resolving a signalling sympathy sickness problem is very likely the same as would be needed to resolve an impaired member problem. Installations that want to preserve past behavior to the greatest extent possible, which is to say, they want to prevent the system from terminating impaired critical members, will need to create and activate an SFM policy with a MEMSTALLTIME specification.

It is NOT possible to completely disable the termination of impaired critical members. However, by specifying a large MEMSTALLTIME value, one can in effect delay the action for so long that it is unlikely to be taken. One would expect one of two things to have happened before the MEMSTALLTIME value expires, either (1) the system resumes normal behavior, or (2) the system is re-IPLed because the “sick but not dead” issues effectively rendered the system unusable.

If you want/need to set an SFM policy MEMSTALLTIME specification, then depending on what you already have set up, you might need to:

- Run the IXCL1DSU utility to create a couple data set that will be used to hold SFM policies
- Run the IXCMIAPU utility to create SFM policies with the desired MEMSTALLTIME value
- Make the SFM CDS available to the sysplex (COUPLExx or SETXCF COUPLE)
- Start the desired SFM policy (SETXCF START,POLICY,TYPE=SFM,POLNAME=xxx)

XES Connector Hang Detection



- Connectors to CF structures need to participate in various processes and respond to relevant events
- XES monitors the connectors to ensure that they are responding in a timely fashion
- If not, XES issues messages (IXL040E, IXL041E) to report the unresponsive connector
- Users of the structure may hang until the offending connector responds or is terminated
 - Impact: sympathy sickness, delays, outages
- Need a way to resolve this automatically ...

88



The XES hang detect function was introduced in OS/390 V1R8 (HBB6608) to report cases when an expected response to a structure-related event is not received in a timely manner. After 2 minutes without a response, XES issues IXL040E or IXL041E to identify the unresponsive connector, the associated structure, the event, and the affected process. Installations often fail to react to these messages, or worse, react by terminating the wrong connector.

IXL040E CONNECTOR NAME: conname, JOBNAME: jobname, ASID: asid
 HAS NOT responsetext
 process
 FOR STRUCTURE strname CANNOT CONTINUE.
 MONITORING FOR RESPONSE STARTED: monddate montime
 DIAG: x x x

IXL041E CONNECTOR NAME: conname, JOBNAME: jobname, ASID: asid
 HAS NOT RESPONDED TO THE event FOR
 SUBJECT CONNECTION: subjectconname.
 process
 FOR STRUCTURE strname CANNOT CONTINUE.
 MONITORING FOR RESPONSE STARTED: monddate montime
 DIAG x x x

Need an automated “backstop” to attempt to resolve the sympathy sickness condition in cases where operators or automation fail to resolve it.

z/OS 1VR12 – CFSTRHANGTIME ...



- CFSTRHANGTIME
 - A new SFM Policy specification
 - Indicates how long the system should allow a structure hang condition to persist before taking corrective action(s) to remedy the situation
- Corrective actions may include:
 - Stopping rebuild
 - Forcing the user to disconnect
 - Terminating the connector task, address space, or system

89



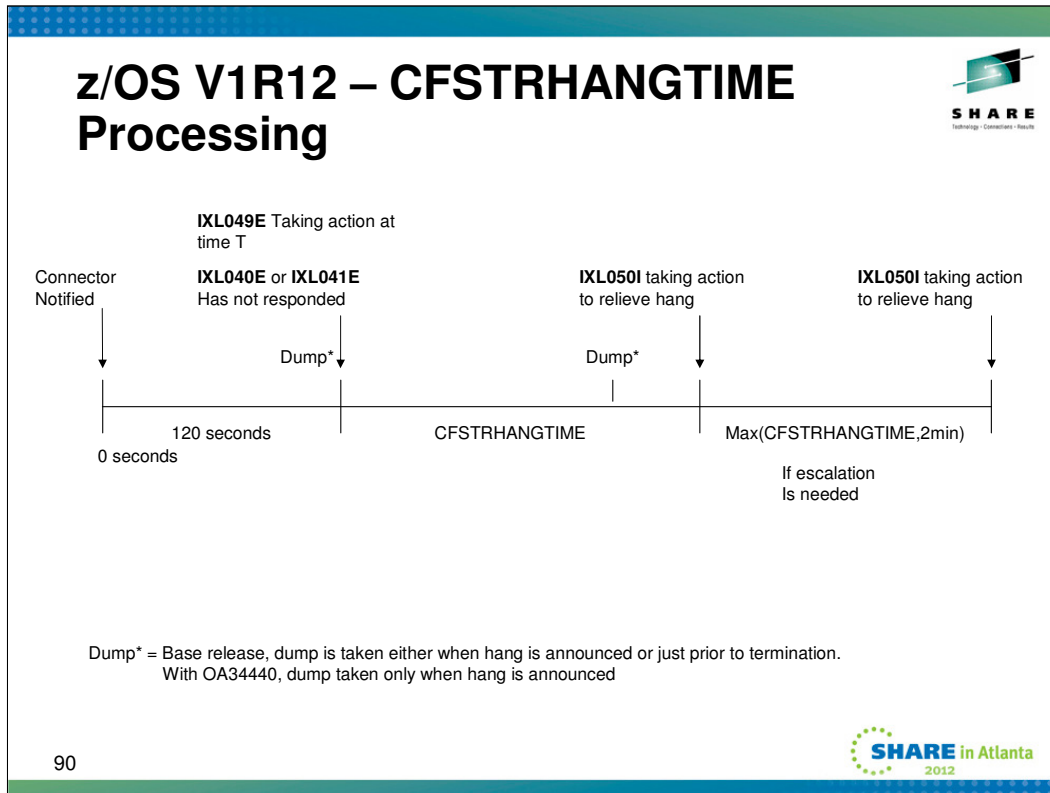
The existing XCF/XES CF structure hang detect support is extended by providing a new CFSTRHANGTIME SFM Policy option that allows you to specify how long CF structure connectors may have outstanding responses. When the time is exceeded, SFM is designed to drive corrective actions to try to resolve the hang condition. This helps you avoid sysplex-wide problems that can result from a CF structure that is waiting for timely responses from CF structure connectors.

The IXCMIAPU policy utility provides a new keyword CFSTRHANGTIME for the SFM policy:

```
DEFINE POLICY NAME(TEST) CONNFAIL(YES) REPLACE(YES)
SYSTEM NAME(*)
ISOLATETIME(0)
WEIGHT(10)
CFSTRHANGTIME(900)
```

The interval specified with the CFSTRHANGTIME keyword begins after a hang is recognized, approximately 2 minutes after the delivery of the event that requires a response. CFSTRHANGTIME(0) means that the system is to take action immediately upon recognizing that the response is overdue. The initial suggestion, documented by way of the new XCF_SFM_CFSTRHANGTIME health check, is to set CFSTRHANGTIME at 5 minutes, but was increased by APAR OA34439 to 15 minutes. This allows time for the installation to evaluate the situation and decide whether to take manual action, and possibly allow the hang to clear spontaneously, while not permitting the hang to persist long enough to cause sysplex-wide problems.

Initial action is taken at expiration of CFSTRHANGTIME interval. If hang persists, escalates to more aggressive actions. The escalation hierarchy begins with the least disruptive actions and progresses to more disruptive actions. Actions include: stop rebuild, stop signaling path (XCF signaling only), force disconnect (XCF signaling only), terminate connector task, terminate connector address space, partition connector system. Each system acts against its own connectors (no system will take action against any other system). No attempt to evaluate causal relationships between multiple events is made.



This chart depicts how XES deals with stalled connectors that are not providing responses in a timely manner.

The connector is notified of an event. If connector does not provide the expected response in roughly two minutes, message IXL040E or IXL041E is issued to indicate that the member has not provided the expected response. Message IXL049E is issued to indicate the means by which the problem is to be resolved, should it persist. If the SFM policy is not active, or the policy specifies (or defaults to) CFSTRHANGTIME(NO), then manual intervention is required (as in the past). If CFSTRHANGTIME is specified to allow automatic action, message IXL049E indicates when that action will occur.

If the CFSTRHANGTIME interval expires and the connector has still not responded, XES issues message IXL050I to indicate the action that is being taken to attempt to relieve the hang condition. In most cases, it is expected that this first action will resolve the problem. However, XES continues to monitor the situation. If the problem persists as long as the CFSTRHANGTIME interval, or two minutes (whichever is longer), XES will escalate to a more “powerful” action. Message IXL050I is issued to indicate what action is being taken. This cycle repeats until the problem is resolved or no further actions apply.

XES will take only one dump per set of monitored events. If monitoring multiple events, any one or any combination of them may have caused or contributed to the hang, but since we can’t analyze those possible relationships there is no point in taking repeated dumps after the hang is first recognized. Once a dump is taken, no further dumps will be taken until all events being monitored have been processed.

If the hang action analysis concludes that automatic action will not be taken to relieve the hang, a dump is taken on the same analysis cycle. Except for the case of policy changes, that would be on the same cycle in which the hang is recognized, which implies that the dump is taken after issuing the IXL040E / IXL041E and IXL049E messages.

If the next anticipated hang relief action is termination of the connector’s task or “soft” termination of the connector’s address space (i.e., task recovery will be allowed to run), the connector’s recovery will presumably take a dump. We expect that an application dump would be more valuable in debugging root cause than a XES dump, so we avoid taking a dump that might inadvertently prevent capture of the application’s dump. Only if we believe that there is sufficient time to capture a XES dump before initiating action will we do so.

If the next anticipated action is not expected to produce a connector dump, XES will take one.


For down-level releases, OA28298 introduced dumping of connector address and data spaces at hang recognition time.

Dump title is tailored to indicate a connector issue:

ABEND=026, REASON=08118001, CONNECTOR HANG: CONNAME=conname, JOBNAME=jobname

APAR OA34440 changed the dump timing so that if the system intends to initiate a corrective action in the future, a dump is taken as soon as the hang is recognized (unless it would interfere with an expected connector dump). The original behavior was to wait until close to the time of automatic action before dumping. Taking the dump closer to the point of the hang provides for better first failure data capture.


z/OS 1.12 – CFSTRHANGTIME ...



New Messages

IXL049E HANG RESOLUTION ACTION FOR CONNECTOR NAME: conname
TO STRUCTURE strname, JOBNAME: jobname, ASID: asid:
actiontext

IXL050I CONNECTOR NAME: conname TO STRUCTURE strname,
JOBNAME: jobname, ASID: asid
HAS NOT PROVIDED A REQUIRED RESPONSE AFTER noresponsetime
SECONDS.
TERMINATING termtarget TO RELIEVE THE HANG.

91


If no SFM policy is active or the policy specifies CFSTRHANGTIME(NO), no hang relief action is taken. If a policy is started, stopped, or changed, the monitor will re-evaluate the required response and possibly reissue IXL049E.

IXL049E may indicate that:

- (1) The system will not take action because (a) there is no SFM policy, (b) the SFM policy requires manual intervention, or (c) the system has tried everything it knows how to do, or
- (2) The system is taking action now (either because the policy specified CFSTRHANGTIME(0) or it was changed in a way that makes the action past due), or
- (3) The system will take action at the time specified in the message.

For IXL049E, *actiontext* is one of:

SFM POLICY NOT ACTIVE, MANUAL INTERVENTION REQUIRED.
SFM POLICY REQUIRES MANUAL INTERVENTION.
SYSTEM IS TAKING ACTION.
SYSTEM WILL TAKE ACTION AT termdate termtime
SYSTEM ACTION UNSUCCESSFUL, MANUAL INTERVENTION REQUIRED

For IXL050I *termtarget* is one of:

REBUILD
SIGNAL PATHS (ATTEMPT 1)
SIGNAL PATHS (ATTEMPT 2)
SIGNAL PATHS (ATTEMPT 3)
CONNECTION
CONNECTOR TASK
CONNECTOR SPACE (WITH RECOVERY)
CONNECTOR SPACE (NO RECOVERY)
CONNECTOR SYSTEM

z/OS V1R12 – CFSTRHANGTIME ...



- Coexistence
 - Toleration APAR OA30880 for z/OS V1R10 and z/OS V1R11 makes reporting of the CFSTRHANGTIME keyword with IXCMIAPU utility possible on those releases.
 - However the capability to take action to resolve the problem is not rolled back to previous releases

92



The support provided by APAR OA30880 allows the IXCMIAPU policy utility initiated from a z/OS V1R10 or V1R11 system to report the CFSTRHANGTIME value specified by a z/OS V1R12 system. It does not permit down-level systems to specify CFSTRHANGTIME when defining an SFM policy, nor does it activate the automatic hang relief function associated with CFSTRHANGTIME. Since the CFSTRHANGTIME specification has no effect on down-level systems, there is no support provided for displaying the CFSTRHANGTIME setting via the DISPLAY XCF,COUPLE command on those systems.

z/OS 1.12 – Support for CFLEVEL 17



- Large CF Structures
 - Increased CF structure size supported by z/OS to 1TB
 - Usability enhancements for structure size specifications
 - CFRM policy sizes
 - Display output
- More CF Structures can be defined
 - New z/OS limit is 2048 (CF limit is 2047)
- More Structure Connectors (CF limit is 255)
 - Lock structure – new limit is 247
 - Serialized list – new limit is 127
 - Unserialized list – new limit is 255

93



z/OS V1.12 on z/Enterprise 196 servers with Coupling Facility Control Code (CFCC) Level 17 supports up to 2047 structures per Coupling Facility (CF) image, up from the prior limit of 1023. This allows you to define a larger number of data sharing groups, which can help when a large number of structures must be defined, such as to support SAP configurations or to enable large Parallel Sysplex configurations to be merged. This function requires the PTF for APAR OA32807; PTFs are also available for z/OS v1.10 and z/OS V1.11. CFRM supports up to 2048 structures, whereas CFLEVEL 17 supports at most 2047. So in actual use, at most 2047 structures could be allocated at one time in any one CF.

z/OS V1.12 on z/Enterprise 196 servers with CFCC Level 17 also supports more connectors to list and lock structures. XES and CFCC already support 255 connectors to cache structures. With this new support XES also supports up to 247 connectors to a lock structure, 127 connectors to a serialized list structure, and 255 connectors to an unserialized list structure. This support requires the PTF for APAR OA32807; PTFs are also available for z/OS V1.10 and z/OS V1.11. CF Level 17 supports 255 connectors to all types of structures. The nature of the z/OS exploitation of those structures requires that the software impose smaller limits than what the CF supports. For lock structures, 8 bits of the 32 byte lock table entry is consumed for “global ownership”, leaving only 247 bits to represent shared interest. For a serialized list, one bit of the one byte lock table entry is used for lock management, so $x'7F' = 127$ is the largest number of connectors that can be represented in the remaining bits of the byte.

z/OS V1.12 supports larger Coupling Facility (CF) structures. The maximum size you can specify for a CF structure is increased from slightly below 100 GB (99,999,999 KB) to 1 TB. The CFRM policy utility (IXCMIAPU) is updated to allow you to specify structure sizes in units of KB, MB, GB, and TB. These changes improve both Parallel Sysplex CF structure scalability and ease of use.

z/OS 1.12 – Support for CFLEVEL 17 ...



- A new version of the CFRM CDS is needed to define more than 1024 structures in a CFRM policy
- May need to roll updated software around the sysplex for any exploiter that wants to request more than 32 connectors to list and lock structures
 - Not aware of any at this point (so really just positioning for future growth)

94



The Couple Data Set (CDS) containing Coupling Facility Resource Manager (CFRM) policies for the sysplex needs to be reversioned in order to allow CFRM policies that define more than 1024 coupling facility (CF) structures.

Only systems running z/OS V1R10 (or later) with the functionality of APAR OA32807 installed can use a CFRM CDS that is formatted to support more than 1024 structure definitions. Once such a CDS is brought into use, down level systems that need to exploit CFRM will not be able to join the sysplex. Since a sysplex-wide outage is required to fall back to a CFRM CDS that does not support more than 1024 structures, it is advisable to delay using this support until you are satisfied that no system in the sysplex will need to run (fall back to) down level code. Versioning the CFRM CDS to support new functions and protocols is the same technique used in the past for message-based protocols, system-managed rebuild, and again for system-managed duplexing. The new version allows CFRM to ensure that all systems can deal with the greater number of structure definitions.

Use the Couple Data Set (CDS) format utility IXCL1DSU to format a CFRM CDS that supports more than 1024 structures by coding the NUMBER keyword with an appropriate value: (see "Setting Up A Sysplex" for complete details):

ITEM NAME(STR) NUMBER(nnnn)

Finally bring the new CFRM CDS into use. For an existing sysplex that is already using CFRM (most likely case), use the SETXCF COUPLE,TYPE=CFRM,ACOUPLE command to define the new CFRM CDS as an alternate, then use the SETXCF COUPLE,TYPE=CFRM,PSWITCH command to make it the primary CFRM CDS. Don't forget to fix the single point of failure by issuing another SETXCF COUPLE,TYPE=CFRM,ACOUPLE command to define another CFRM CDS that supports at least as many structure definitions as an alternate for redundancy. For an existing sysplex that is not using CFRM (not likely), use the SETXCF COUPLE,TYPE=CFRM,PCOUPLE command to define an appropriate CFRM CDS as the primary. If IPLing the first system into the sysplex, define the CFRM CDS (primary and alternate) in the COUPLExx parmlib member.

The Administrative Utility (IXCMIAPU) can then be used to define CFRM policies with as many structure definitions as the CDS supports.

Enabling support for more than 32 connectors to lock and list structures requires software upgrades. The new version of the CFRM CDS is not needed for this aspect of z/OS 1.12 support. All systems in the sysplex that need to connect to a structure that can have more than 32 connectors will need to have z/OS V1R10 (or later) with the PTFs for APAR OA32807 installed. In addition, the exploiters are required to code a new MAXCONN keyword on the invocation of the IXLCONN macro that is used to connect to the structure.

z/OS 1.12 – Support for CFLEVEL 17 ...



- z/OS requests non-disruptive CF dumps as appropriate
- Coherent Parallel-Sysplex Data Collection Protocol
 - Exploited for duplexed requests
 - Triggering event will result in non-disruptive dump from both CFs, dumps from all connected z/OS images, and capture of relevant link diagnostics within a short period
 - Prerequisites:
 - Installation must ENABLE the XCF function DUPLEXCFDIAG
 - z/OS 1.12
 - z/OS 1.10 or 1.11 with OA31392 (IOS) and OA31387 (XES)
 - Note that full functionality requires that:
 - z/OS image initiating the CF request reside on a z196
 - CF that “spreads the word” reside on a z196

95



The DUPLEXCFDIAG optional function is defined to allow an installation to control the use of the “Coherent Parallel-Sysplex Data Collection Protocol”, aka the “link diagnostics” protocol. By default, the protocol is disabled, and it is expected that an installation would only enable it if it experiences duplexing issues. If the DUPLEXCFDIAG function is disabled, z/OS will not request use of the protocol when issuing duplexed CF requests. DUPLEXCFDIAG is enabled and disabled using the same parmlib (COUPLExx) and command (SETXCF FUNCTIONS) interfaces as all other optional XES / XCF functions.

Even if the function is ENABLED, the protocol is requested only if both CFs are at or above CFLEVEL 17, since there’s no point collecting link data if we aren’t going to get the whole picture. The protocol relies on the CF to capture non-disruptive dumps and to propagate the request for data to the peer CF and connected z/OS images.

z/OS must set parameters in the request that is sent to the CF to indicate that this form of data collection is desired. Similarly, if a CF detects a triggering event, it must set parameters in the notifications that are sent to the peer CF and the connected z/OS images. These parameters are interpreted by the links. Only the z196 side of the links has the ability to process these parameters. Thus, full functionality of the coherent data collection would only be available if the z/OS image initiating the request resides on a z196, and if the CF that detects the triggering event and propagates the request for data collection also resides on a z196. With the appropriate releases and/or PTFs installed, the z/OS images that do not reside on a z196 can generate the appropriate dumps when asked, but they would not be able to initiate coherent data collection.

z/OS 1.12 Health Checks



- **XCF_CF_PROCESSORS**
 - Ensure CF CPU's configured for optimal performance
- **XCF_CF_MEMORY_UTILIZATION**
 - Ensure CF storage is below threshold value
- **XCF_CF_STR_POLICYSIZE**
 - Ensure structure SIZE and INITSIZE values are reasonable

96



New health checks for the Parallel Sysplex components, XCF and XES, are included in z/OS 1.12. These checks can help you correct and prevent common sysplex management problems.

XCF_CF_PROCESSORS

Raises an exception if a CF is not configured with all CPs dedicated. The IBM supplied check raises an exception if any CF in use by the sysplex is configured with one or more shared CP's. To obtain maximum CF performance and throughput, a coupling facility should be configured completely with dedicated processors instead of shared processors. The installation can specify a check parameter to exclude designated CF's from being considered by the check. For example, one might elect to exclude a CF that is using shared CP's because it is part of a test configuration.

XCF_CF_MEMORY_UTILIZATION

Raises an exception if the CF storage usage exceeds the indicated threshold. The IBM supplied check raises an exception if the storage utilization exceeds 60%. The percentage of memory utilization in a coupling facility should not approach an amount so high as to prevent the allocation of new structures or prevent the expansion of existing structures. The installation can specify a check parameter to designate some other threshold.

XCF_CF_STR_POLICYSIZE

Raises an exception if the CFRM policy definition for a structure has size specifications (SIZE and INITSIZE) that can either (a) cause CF storage to be wasted, or (b) make the structure unusable, or (c) prevent the structure from being allocated. Specifying different INITSIZE and SIZE values provides flexibility to dynamically expand the size of a structure for workload changes, but too large a difference between INITSIZE and SIZE may waste coupling facility space or prevent structure allocation.

z/OS 1.12 Health Checks ...



- XCF_CDS_MAXSYSTEM
 - Ensure function CDS supports at least as many systems as the sysplex CDS
- XCF_CFRM_MSGBASED
 - Ensure CFRM is using desired protocols
- XCF_SFM_CFSTRHANGTIME
 - Ensure SFM policy using desired CFSTRHANGTIME specification

Initially complained if more than 300 (5 minutes).
 APAR OA34439 changed it to 900 (15 minutes)
 to allow more time for operator intervention and
 more time for all rebuilds to complete after losing
 connectivity to a CF

97



XCF_CDS_MAXSYSTEM

Raises an exception when a function couple data set (CDS) is formatted with a MAXSYSTEM value that is less than the MAXSYSTEM value associated with the primary sysplex CDS. A “function CDS” is any CDS other than the sysplex CDS. For example, a function CDS might contain CFRM policies or SFM policies. If a function CDS does not support at least as many systems as the sysplex CDS, a new system might not be fully functional when it joins the sysplex (if it can join at all).

XCF_CFRM_MSGBASED

Raises an exception if CFRM is not configured to exploit the desired structure event management protocols. The IBM supplied check raises an exception if CFRM is not configured to exploit “message based” protocols. The CFRM “message based” protocols were introduced in z/OS V1R8. Use of this protocol reduces contention on the CFRM policy CDS, which can significantly reduce the recovery time for events that trigger CFRM activity against lots of structures (such as loss of a CF or a system). The installation can specify a check parameter to indicate which CFRM protocol is desired, “message based” or “policy based”.

XCF_SFM_CFSTRHANGTIME

Raises an exception if the active SFM policy is not consistent with the indicated CFSTRHANGTIME specification designated by the check. The IBM supplied check will raise an exception if the SFM policy specifies (or defaults to) CFSTRHANGTIME(NO), or if the SFM policy specifies a CFSTRHANGTIME value greater than 900 seconds (was 300 seconds prior to APAR OA34439). The installation can specify a check parameter to indicate the CFSTRHANGTIME value that the installation wants to use. The check will then raise an exception if the SFM policy is not in accordance with the check parameter.

z/OS 1.12 Auto-Reply



- Fast, accurate, knowledgeable responses can be critical
- Delays in responding to WTOR's can impact the sysplex
- Parmlib member defines a reply value and a time delay for a WTOR. The system issues the reply if the WTOR has been outstanding longer than the delay
- Very simple automation
- **Can be used during NIP !**

98



Many installations no longer have operators closely monitoring the system waiting to immediately reply to a WTOR. Operators typically do not have authority, experience, or system understanding to make their own decision on what to reply for uncommon WTORs. Customers have told us that it is reasonable to assume a WTOR may take at least **30-45 minutes** to be answered. So it is no longer feasible to expect fast, accurate, knowledgeable answers from system operators. Some WTORs are so infrequent that they are not automated and operators may never have seen them before. Reply delays can affect all systems in sysplex.

In z/OS V1.12, a new Timed Auto Reply function enables the system to respond automatically to write to operator with reply (WTOR) messages. This new function is designed to help provide a timely response to WTORs and help prevent delayed responses from causing system problems.

The Timed Auto Reply Function allows you to specify message IDs, timeout values, and default responses in an auto-reply policy, and to be able to change, activate, and deactivate autoreply with operator commands. Also, when enabled, it starts very early in the IPL process, before conventional message-based automation is available, and continues unless deactivated. You can also replace or modify an IBM-supplied auto-reply policy in a new AUTOR00 parmli member. This new function is expected to help provide a timely response to WTORs and help prevent delayed responses from causing system problems.

z/OS 1.12 Auto-Reply



- For example:

```
IXC289D REPLY U TO USE THE DATA SETS LAST USED  
FOR typename OR C TO USE THE COUPLE DATA SETS  
SPECIFIED IN COUPLExx
```

- The message occurs when the couple data sets specified in the COUPLExx parmlib member do not match the ones in use by the sysplex (as might happen when the couple data sets are changed dynamically via SETXCF commands to add a new alternate or switch to a new primary)
- Most likely always reply “U”

99



From a sysplex perspective, Auto-Reply will likely prove useful for messages to which the operator must reply while the system is IPLing. May prove quite useful for disaster recovery testing.

z/OS 1.12 - XCF Programming Interfaces



- IXCMSGOX
 - 64 bit storage for sending messages
 - Duplicate message toleration
 - Message attributes: Recovery, Critical
- IXCMSGIX
 - 64 bit storage for receiving messages
- IXCJOIN
 - Recovery Manager
 - Critical Member
 - Termination level

100



Two new services based on existing XCF signaling services are introduced to support the use of 64-bit addressable virtual storage message buffers and associated input and output parameters. The two new services, IXCMSGOX and IXCMSGIX, are the 64-bit counterparts of the existing IXCMSGO and IXCMSGI services, which are used by XCF group members to send and receive messages. These new services make it easier for exploiters to achieve virtual storage constraint relief by removing the need to copy message buffers and associated storage structures from 64-bit addressable virtual storage to 31-bit storage and back.

IXCMSGOX allows the sender to indicate that the target(s) can tolerate receiving duplicate copies of the message, which allows XCF to resend the message sooner if there should be a signal path failure while the message is being sent. Other new message attributes include “recovery” which is used to identify signals that are involved in recovery processes, and “critical” which indicates that the message is critical to the exploiter.

The IXCJOIN service, which is invoked to become a member of an XCF group, has some new keywords that enable the member to indicate that it is a “recovery manager” that performs a sysplex-wide recovery process, or that it is a “critical member” that is to be terminated by XCF if it appears to be unresponsive. The “termination level” allows the member to indicate the scope at which it wants to be terminated (task, address space, or system) when XCF decides to do so.

Appendix – z/OS V1R11



- System Default Action
- XCF FDI Consistency

101



These topics were omitted from the presentation due to time restrictions.

z/OS 1.11 - System Default Action



- SFM Policy lets you define how XCF is to respond to a Status Update Missing condition
- Each system “publishes” in the sysplex couple data set the action that is to be applied by its peers
- The system “default action” is published if:
 - The policy does not specify an action for it
 - There is no SFM policy active
- Prior to z/OS 1.11, the “default action” was PROMPT
- With z/OS 1.11, the system default action is ISOLATETIME(0)

102



If you want to preserve past default behavior of PROMPT for a z/OS 1.11 system, you would need to take a migration action to define and activate an SFM policy that explicitly specifies PROMPT. However, this is not recommended since best practice is ISOLATETIME(0). If you already explicitly specify PROMPT, consider changing to use the best practice specification of ISOLATETIME.

z/OS 1.11 - System Default Action



- The resulting behavior for system “default action” depends on who is monitoring who:
 - z/OS 1.11 will isolate a peer z/OS 1.11
 - z/OS 1.11 will PROMPT for lower level peer
 - Lower level system will PROMPT for z/OS 1.11
- D XCF,C shows what the system *expects*
 - *But it may not get that in a mixed sysplex*
- Note: z/OS 1.11 may fence even if action is PROMPT
 - Lower level releases performed fencing only when the system was taking automatic action to remove the system (ISOLATETIME)

103



The SFM Policy lets you define how XCF is to respond to a Status Update Missing condition. Each system “publishes” in the sysplex couple data set the action that is to be applied by its peers. The system “default action” is published if either (a) the policy does not specify an action for it, or (b) there is no SFM policy active. Prior to z/OS 1.11, the “default action” was PROMPT, which causes the peer systems to prompt the operator (with message IXC402D) when the system appears to be “status missing”.

IBM suggests specifying or defaulting to ISOLATETIME(0) to allow SFM to fence and partition a failed system without operator intervention and without undue delay. As of z/OS V1R11, the system default will be in accord with this suggestion.

If a system enters a status update missing condition and there is no active SFM policy, the monitoring system will take the system default against the failed system. This means if the monitoring system is a pre-z/OS V1R11 system, it will use the old system default and prompt the operator. If the monitoring system is a z/OS V1R11 system, it will use the system default of the failed system. The D XCF,C command shows the isolate action that the system expects, but the monitoring system may use a different action if no action is specified in the SFM policy or if an SFM policy is not active. In a sysplex that does not have an SFM policy and has images running z/OS 1.10 or earlier, the z/OS 1.11 and later systems may display N/A for the expected action because they don't really know what will happen. The result depends on which system processes the partitioning request.

Default action of ISOLATETIME(0) is not guaranteed. z/OS 1.11 observing a peer z/OS 1.11 knows the new default should be ISOLATETIME(0). But a z/OS 1.10 or 1.9 observing a z/OS 1.11 will continue to treat the default as PROMPT. A z/OS 1.11 observing a z/OS 1.9 or 1.10 system knows that their default action was PROMPT, and continues to honor that old behavior. In order to actually do an ISOLATETIME(0), one either needs to be able to fence the system or use BCPII to reset the system. **So if SFM is unable to do either of those, it still reverts to PROMPT.**

z/OS 1.11 - XCF FDI Consistency



- Enforces consistency between the system Failure Detection Interval (FDI) and the excessive spin parameters
- Allows system to perform full range of spin recovery actions before it gets removed from the sysplex
- Avoids false removal of system for a recoverable situation

Helps prevent false SFM removals


104

XCF failure detection interval coordination with spin time

Enforce consistency between XCF FDI interval and the effective excessive spin time actions and intervals, by automatically adjusting the FDI to be at least as high as that implied by excessive spin specifications

In z/OS V1.11, XCF design is changed to automatically adjust the failure detection interval (FDI) to use for systems in the sysplex when needed. The system's effective FDI is now designed to be the longer of the two intervals resulting from the FDI you specify and a value based on the system's excessive spin parameters, making the system's processing of excessive disabled spin conditions, the sysplex's handling of missing system heartbeats, and the initiation of sysplex partitioning to remove unresponsive systems more consistent. Also, a new way to specify an operator notification (OPTNOTIFY) relative to the effective FDI is provided, so that you no longer need to calculate the sum of spin loop timeouts to specify the operator notification interval.

z/OS 1.11 - XCF FDI Consistency



```

IXC357I 15.12.46 DISPLAY XCF Effective Values E SYS=D13ID71
SYSTEM D13ID71 DATA
  INTERVAL  OPNOTIFY  MAXMSG  CLEANUP  RETRY  CLASSLEN
    165      170      3000    60         10     956

  SSUM ACTION  SSUM INTERVAL  SSUM LIMIT  WEIGHT  MEMSTALLTIME
    PROMPT          165          N/A         0         N/A

  PARMLIB USER INTERVAL: 60
  DERIVED SPIN INTERVAL: 165
  SETXCF  USER OPNOTIFY: + 5

< - - - snip - - - >
OPTIONAL FUNCTION STATUS:
FUNCTION NAME      STATUS      DEFAULT
DUPLXCF16         ENABLED    DISABLED
SYSSTATDETECT     ENABLED    ENABLED
USERINTERVAL      DISABLED   DISABLED
  
```

User FDI
 Spin FDI
 User OpNotify
 - Absolute
 - Relative

Switch

This slide shows relevant output from the DISPLAY XCF,COUPLE command. The effective FDI and OpNotify value are reported as in past releases. A new section reports user specified FDI, derived spin FDI, and user specified OpNotify value as well as the source from which the current value was derived (COUPLExx parmlib, SETXCF command, system default). The FUNCTION section now reports the state of the new USERINTERVAL switch.

The **Failure Detection Interval (FDI)**, is the amount of time that a system can appear to be unresponsive (status update missing) before XCF is to take action to resolve the problem. The **User FDI** is an FDI value explicitly specified by the user, either directly (via COUPLExx parmlib member or SETXCF COUPLE command) or indirectly (through cluster services interfaces – IXCCROS macro). The **Spin FDI** is an FDI value derived by XCF from the excessive spin parameters (spin loop timeout value and the number of excessive spin actions (such as SPIN, TERM, ACR)). The **Effective FDI** is the FDI value that is being used for the system. If the USERINTERVAL switch (FUNCTIONS) is DISABLED, the effective FDI = max(user FDI, spin FDI). If USERINTERVAL is ENABLED, the effective FDI = user FDI. By default, the effective FDI will be the larger of the user FDI and the spin FDI. If the installation really wants the smaller user FDI to be the effective FDI, the USERINTERVAL switch must be ENABLED to force XCF to use it.

User OpNotify is the operator notification interval explicitly specified by the user, either directly (via COUPLExx parmlib member or SETXCF COUPLE command). OpNotify determines when XCF should alert the operator about a system that is unresponsive. OpNotify can now be **relative** to the effective FDI (ie, a delta). In the past, it was always an **absolute** that had to be greater than or equal to the (effective) FDI. So if one wanted to change one of the values, one might in fact have to change them both (and in a particular order) so as to maintain the required relationship. With a relative OpNotify value, the system automatically maintains the relationship. If the effective FDI changes, the effective OpNotify value changes as well. The **Effective OpNotify** is the OpNotify value being used by the system. The system ensures that effective FDI is always less than or equal to the effective OpNotify value.

If the installation changes the excessive spin parameters, sets a new user FDI value, or changes the USERINTERVAL switch, the effective values are recomputed and then written to the sysplex CDS to make them visible to the rest of the sysplex (via status update processing).