“Run-it-Back” – Db2 for z/OS All new “2023 SWAT Tales”

Tridex Db2 z/OS

Thursday November 30, 2023

Anthony Ciabattoni aciabattoni@ibm.com

Agenda

- Db2 preventative maintenance considerations
- Local Db2 Backup, Corruption Scope Identification and Recovery
- Continuous Availability
- HiperDispatch
- Growth in CPU time/transaction as CPU busy increases
- “Self Healing”
- Questions

Db2 Preventative Maintenance

Db2 preventative maintenance considerations

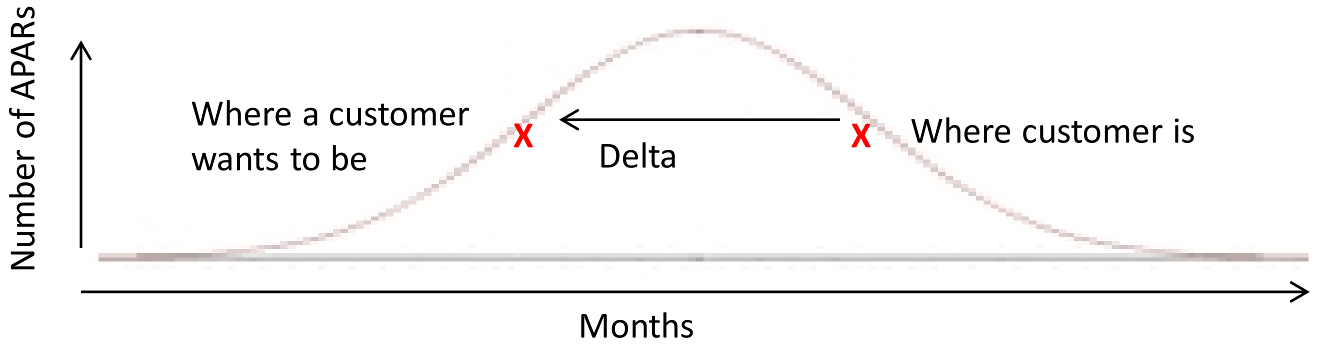
- Introduction
  - The process of applying or not applying Db2 PTF(s) is often-times one of the most important tasks performed by a Db2 professional
    - The result of applying/not applying an APAR can result in …
      - Db2 crash
      - Data corruption
      - Incorrect output (INCORROUT)
      - Etc.
  - Technical resources commonly have developed processes to apply routine maintenance but have not created exception process to apply for critical PEs or HIPER PTF(s)
  - Maintenance frequency, decision processes and exceptions are not effectively communicated to management/executives
    - The end results supports a “blame” culture, the technician is always wrong

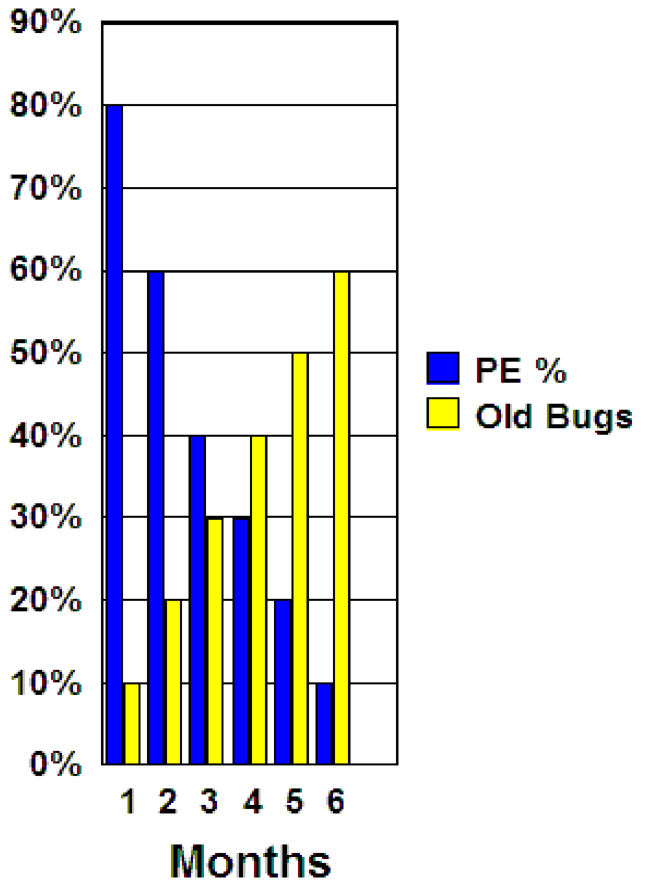
Db2 preventative maintenance considerations …

- Common Problems
  - Too many customers are very back level on preventative service
  - No HIPERs or PE fixes applied since the last preventative service upgrade
  - High profile production incidents could have been avoided by missing HIPER
  - **Unaware of missing new HIPERs, PTFs in error (PE) and corresponding exposures**
  - No insight into security vulnerabilities
  - ‘Fix-on-failure’ culture introduces the probability of long prerequisite chain when having to apply emergency corrective service
  - Not exploiting Db2 Data Sharing technology to avoid planned outages and remove dependency on change windows
  - Delay in applying Db2 serviceability enhancements to prevent outages
  - Delay in exploiting new availability functions

Db2 preventative maintenance considerations …

- Applying preventive maintenance can and will avoid outages
  - Up to 20% of multi-system outages could have been avoided by regularly installing 'critical' (e.g., HIPER and PE fixing) PTFs
- Executing a preventive maintenance process requires an understanding of trade-offs
  - Position on the adoption 'bell curve'
  - Problems encountered vs. problems avoided
  - Potential for PTF in Error (PE)





Db2 preventative maintenance considerations …

- Maintenance trade-off
  - Achieving the highest availability depends on a having an adaptive preventive maintenance process that is adjusted based on …
    - Attitude to risk in changing environment and exploiting new Db2 releases and functions
      - Moving up the adoption curve of a Db2 release should drive more frequent drops of preventative service
        - Also applies to aggressive use of new features
    - Experience over previous 12-18 months
      - Too many PEs should drive less aggressive preventative service apply
      - Too many problems and repeat problems where the fixing PTF is readily available should indicate that more frequent drops of preventative service should be applied
- Recommendations
  - Change Management process should balance the risk of making 'no change' vs. making 'a change'
  - Apply preventative maintenance every 3 months
    - *Sample strategy* based on two 'major' and two 'minor' releases
      - Refresh of the base every 6 months ('major')
        - Each base upgrade should be based on latest quarterly RSU or quarterly +1 month as a base
      - In addition, two mini packages covering HIPERs and PEs in between ('minor')
    - Early adopters of new releases and/or new functions should be more aggressive about applying preventative service

Db2 preventative maintenance considerations …

- Enhanced HOLDDATA and REPORT ERRSYSMODS are a key element for best practice
  http://service.software.ibm.com/holdata/390holddata.html
  - Implement a continuous weekly process to look for high impact HIPERs, PEs and security vulnerability APARs during the package roll out period and for the life of the package running in production
  - Recommended best practice to support the above objective is to have multiple SMP/E representations to match various maintenance levels
    - At a minimum recommend using three target zones
      - Production, production -1, new maintenance package
    - Benefits
      - Continually receive Enhanced HOLDDATA and associated PTFs into a single SMP/E global zone that can be used to independently run REPORT ERRSYSMODS for each environment including production
      - Provides flexibility and simplicity to support multiple drops per year including emergency fixes
  - Produce and review a weekly Enhanced HOLDDATA summary report against production and new maintenance package zones
    - Includes the fixing PTF number when the PTF is available
    - Includes HIPER reason flags
      - IPL, DAL (data loss), FUL(major function loss), PRF (performance), PRV (pervasive)
    - Assign a risk and exposure of the identified HIPER or PTF now in error
      - Identify and implement any potential operational bypasses
      - Expedite critical fixes to production after 1-2 weeks in test
      - Others can be deferred until the next major or minor maintenance drop

Db2 preventative maintenance considerations …

- Common problems - PEs
  - As a new package is built, PEs (and any blocked maintenance) are systematically taken off
  - If no PE fixers are added to the package, it has the potential to leave the package very back-leveled and/or increase operational exposure to serious problems if a large number of HIPERs or even a small number of very critical ones are blocked

- Recommendations
  - Proactively monitor for PE fixers and new PEs on a weekly basis as you are building the package and during the roll, up to ~1 month before production
  - Apply PE fixers as soon as available + any RSU maintenance that can now be unlocked
  - Need a judgement call around remaining PEs that are not yet resolved or have emerged
    - If the PEs are blocking significant HIPER maintenance (large number and/or 'vicious' ones) → do not simply take them off, but look at options
      - Wait for the PE fixer, or look for an operational bypass to get the PE on or to avoid hitting the problems covered by the HIPERs
      - Do not hesitate to open PMRs to get additional information to make a well-informed decision

Local Db2 Backup, Corruption Scope Identification and Recovery

Db2 recovery background

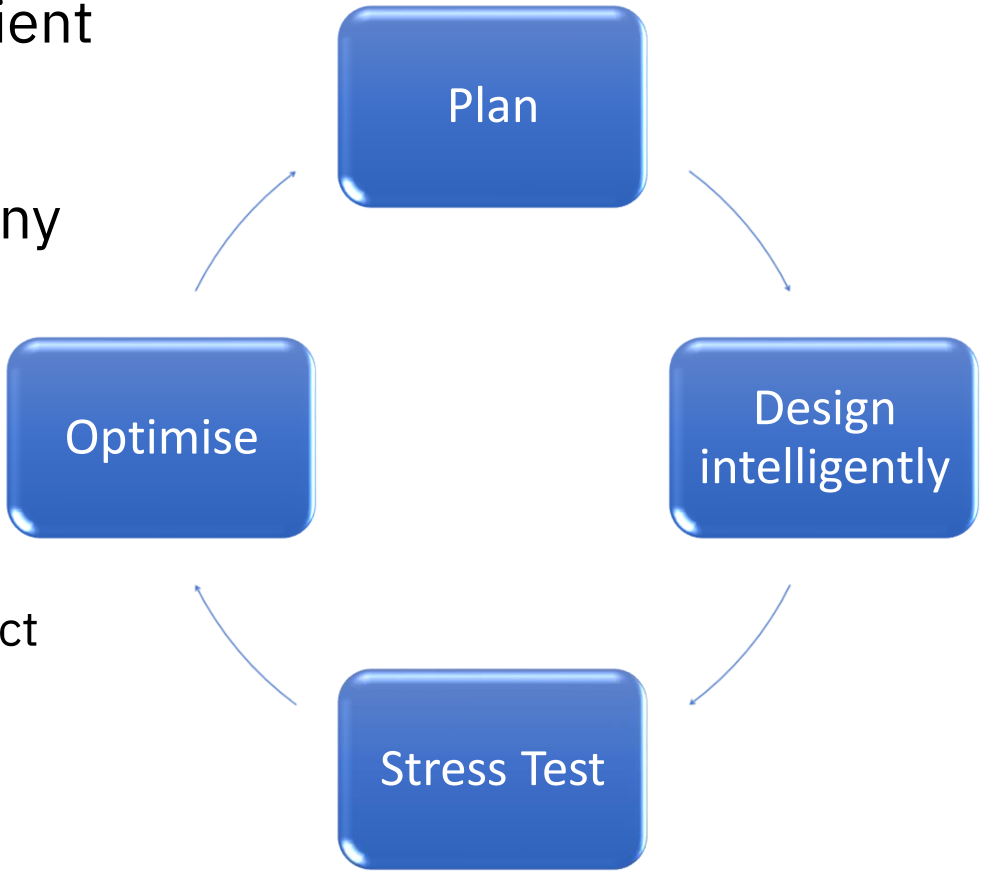
- Db2 log-based recovery of *multiple objects* may be required when…
  - Catastrophic DASD subsystem failure and no second copy
  - Plan B for disaster recovery
    - Mirror is damaged/inconsistent
    - Bad Disaster Restart e.g., using stale CF structures in data sharing
  - Data corruption at the local site caused by…
    - 'Bad' application program
    - Operational error
    - Db2, IRLM, z/OS, third-party product code failure
    - CF microcode failure, DASD subsystem microcode failure
- Scope of the recovery may be more or less extensive
  - One application and all associated objects
  - Part of the system (including a random list of objects across multiple applications)
  - Or, in the worst case, the 'whole world'

Db2 recovery background …

- **Db2 log-based recovery of multiple objects is a very rare event …**

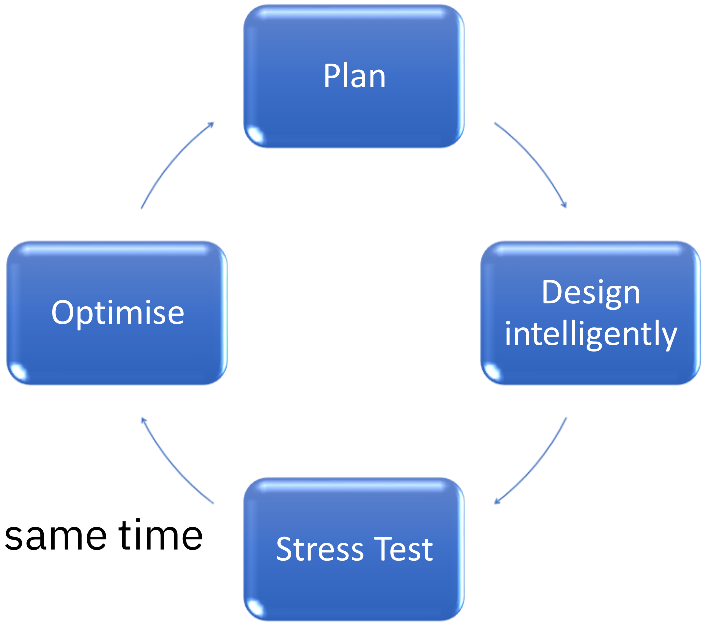
  … but statistically, it is more frequent than a true DR event (flood, fire, etc.)

- Taking regular backups is necessary but far from sufficient for anything beyond minor application recovery
- If not prepared, practiced and optimized, will lead to extended application service downtimes – possibly many hours to several days
  - Things to consider
    - Are my procedures up to date?
    - Configuration changes? Db2 release?
    - Are the archive logs and image copies on DASD, VTS or physical tape?
    - Are image copies and recovery jobs created based on object priority?
      - How long will the "recover" take?
    - Are all my objects backed up?
    - If not practiced "what do you not know?"



High performance multiple object recovery

- Common issues
  - Lack of planning, intelligent design, optimization, practice & maintenance
  - No prioritized list of application objects and inter-dependencies
    - Limited use of Db2 referential integrity
    - Data dependencies and integrity management are buried in the applications
    - Heavily dependent on application knowledge and support
  - Procedures for taking backups and executing recovery compromised by lack of investment in technical configuration
  - Backup and recovery procedures have not been addressed for years
  - Use of tape including VTS *("Identity Crisis")*
    - Cannot share tape volumes across multiple jobs
    - Relatively small number of read devices
    - Concurrent recall can be a serious bottleneck
    - Even though VTS has a disk cache, it is known to z/OS as tape device
      - Same serialization characteristics as all tape devices
      - A single virtual volume cannot be shared by different jobs or systems at the same time

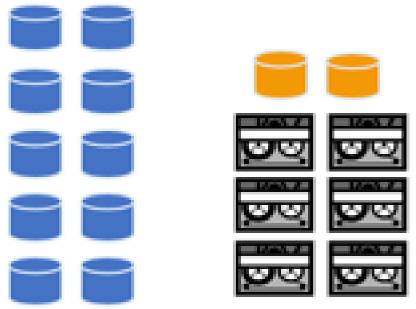


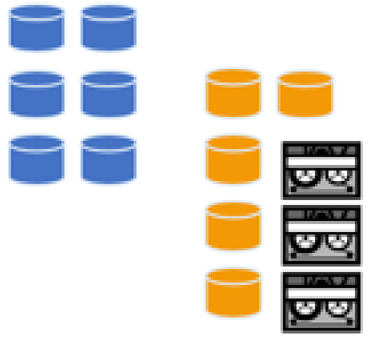
Db2 logging environment

- Common Problems
  - Wrapping active log datasets too frequently (< 6 hours) and not proactively monitoring
  - Writing archive logs directly to tape or VTS
  - If a Global or Metro mirror configuration is being used …
    - Potential "holes/gaps" in active/archive logs at target location
- Risks
  - Wrapping of the active log pairs too frequently can expose availability concerns
    - Small window of time to react if there are issues with archiving the active logs
  - Serious archive log contention during parallel recovery
  - VTS replication latency can result in "holes" in the Db2 active log at target location
    - Prevents the ability to recovery through a LOG NO event
- Recommendations
  - Increase the size of the active log configuration to always hold at a minimum 6 hours of recovery log data, if possible 24 hours is preferable
    - Standardize on a uniform dataset size of up to 4GB-1 bytes → 768GB
    - Add in additional preformatted log pairs (if needed, max 93 pair), will prevent the ability to dynamically allocate additional logs
    - This position must be proactively *monitored and maintained*
    - Db2 13 enables the ability to delete active log datasets, benefits include:
      - Online delete and add of active log datasets for larger sizes
      - Encryption support, online deallocating/allocating encrypted logs i.e., can be required every 2-3 years
      - Move active log datasets to alternative storage devices (storage upgrade)
  - Write both copies of each archive log pair to DASD
    - Keep LOGCOPY1 on DASD for at least 48 hours (must be maintained going forward)
    - Can immediately migrate LOGCOPY2 away to VTS using DFSMShsm

Db2 logging environment …

- Design for *recovery performance* …
  - Keep at least 48h of recovery log data on DASD





Option #1: Over-configure the active log pairs (number/size)
Write archive log COPY1 and COPY2 to DASD but they can be migrated to tape/VTS at any time

Pros: Optimal log read performance with automatic load balancing for reads between active log COPY1 and COPY2,

Db2 12 increases capacity to 93x768GB

Option #2: Keep archive log COPY1 on DASD for 48-72h before migrating it to tape/VTS – archive log COPY2 can be migrated to tape/VTS at any time

Pros: Good log read performance from archive on DASD, potential for less DASD requirements than Option 1

  - Be ready to extend the amount of recovery log beyond what is available on DASD
    - Set BLKSIZE=24576 to optimise reads on DASD
    - Prepare a procedure to copy archive logs from tape or VTS to DASD

Mass application recovery

- Recommendations
  - Agree on a prioritized list of business-critical applications
  - Keep a list of all related data required by these applications
    - Dependencies across application domains
    - Including non-Db2 data
  - Critical information needed during a recovery event
    - Objective: Bring back critical application services as soon as possible
    - Without these lists, either have to wait for the whole world to be recovered, or take a risk in bringing back some of the application services earlier
    - Should not rely exclusively on application expertise
  - Modernized existing Db2 image copy process to take advance of more efficient backup techniques and drive the solution based on Recovery Time Objective (RTO) per application
    - Backup strategy does not need to be a “one sized fits all” solution
    - Identify tablespace objects that are candidates for alternative methods
      - Dataset-level FlashCopy image copy e.g., speed/size/activity
      - Incremental image copy process
        - Full weekly tablespace copies writing to VTS
        - Nightly incremental image copies written to DASD
    - When writing any form of backup to DASD use HSM to migrated the datasets to VTS
      - Essential to recall FlashCopy image copy prior to recovery
        - Otherwise, IBM RECOVER utility will skip the image copy
  - At the first sign of potential data corruption develop a process to stop the rolling away of recovery assets

Data integrity checking

- Introduction
  - Once data corruption has occurred, it is essential to understand the scope of the corruption prior to developing a recovery strategy
    - One application and all associated objects
    - Part of the system (including a random list of objects across multiple applications)
    - Or, in the worst case, the 'whole world'
  - CHECK utilities are critical diagnosis tools in the event of data corruption
    - Identify objects that need repairing/recovering and assess the extent of the damage
      - Drive decision making process e.g., fix forward vs data recovery
  - Majority of installations do not proactive procedures are in place to non-disruptively determine which objects are damaged and the degree of the damage
  - Customers potentially are not set up to run the CHECK utilities non-disruptively
    - Cannot take advantage of FlashCopy unless directing the CHECK utilities to use a pool of volumes outside of Global Mirror for the creation of the shadow objects
    - ZPARM CHECK_FASTREPLICATION = PREFERRED would allow the CHECK SHRLEVEL CHANGE utilities to use 'standard' I/O to create the shadow objects, which could result in an elongated interference with the applications
- Risks
  - Interference with updating applications during accidental use of CHECK SHRLEVEL CHANGE utilities
  - Delays in detecting objects with data inconsistencies and the degree thereof

Data integrity checking …

- Recommendations
  - As a defensive measure, set ZPARM CHECK_FASTREPLICATION = REQUIRED to prevent accidental use of CHECK utilities with SHRLEVEL CHANGE
  - Exploit dataset-level FlashCopy to run the CHECK utilities non-disruptively
    - Make sure that multi-volume datasets are within the same physical storage subsystem
      - Datasets can be spread over many LCUs so long as it is not across multiple physical storage subsystems
  - IBM Global Mirror (GM) primary can now be a FlashCopy target
    - FlashCopy target data will be sent over the replication links to GM secondary
    - However, there is potential to affect the recovery point objective (RPO) of GM when there are many concurrent FlashCopy operations
      - If degraded RPO is a problem
        - Carve out a pool of volumes outside of Global Mirror for temporary shadow copies
          - The volumes could be part of Metro Mirror, but then must set ZPARM FLASHCOPY_PPRC = REQUIRED to use Remote Pair FlashCopy and avoid going out of full duplex on the Metro Mirror pairs
        - Use ZPARM UTIL_TEMP_STORCLAS to specify a storage class mapped to the pool of target volumes outside of Global Mirror (or use the DFSMSdss FlashCopy Batch Protection feature)
    - Note: Dataset FlashCopy will fail if individual dataset should span multiple DASD storage subsystems

Data integrity checking …

- Recommendations …
  - Exploit the Disaster Recovery (DR) infrastructure to avoid any operational impact on the production system
    - Take advantage of the space for tertiary copy used for DR testing
    - Take a new snap tertiary copy
    - If there are significant number of large objects to be checked must turn on additional engines on the DR LPAR(s)

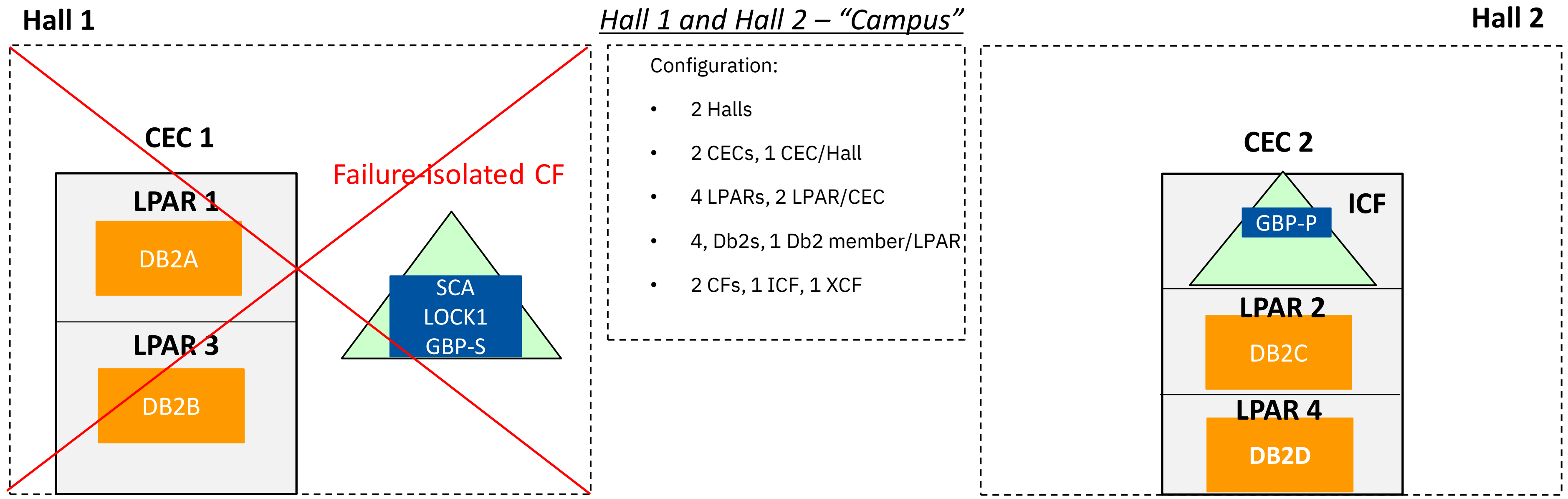
Mass application recovery …

- Recommendations …
  - After the scope of the data corruption has been quantified, build recovery jobs that exploit the capacity of the entire Db2 data sharing group
    - Maximum parallelism in the RESTORE phase
      - For partitioned tablespaces, use parallelism by part
        - LISTDEF utility statement with the PARTLEVEL option will build a list of partitions for an object and automatically handle partitions that are added or pruned
      - Use PARALLEL for parallel processing from image copies on DASD
      - Use PARALLEL(n) TAPEUNITS(n) for image copies stacked on tape
    - Optimal use of fast log apply (FLA)
      - Db2 sets internally to 510MB
      - Schedule up to 51 RECOVER jobs per Db2 subsystem
      - RECOVER a list of objects rather than individual objects
        - But no more than 98 objects per RECOVER job for best results (1 partition = 1 object)
        - 20-30 objects per RECOVER job seems to be optimal for FLA use
        - Single pass of the recovery log for all objects in the list
      - Spread the jobs across all Db2 data sharing members

Continuous Availability

Achieving Continuous Availability

- Parallel Sysplex configuration



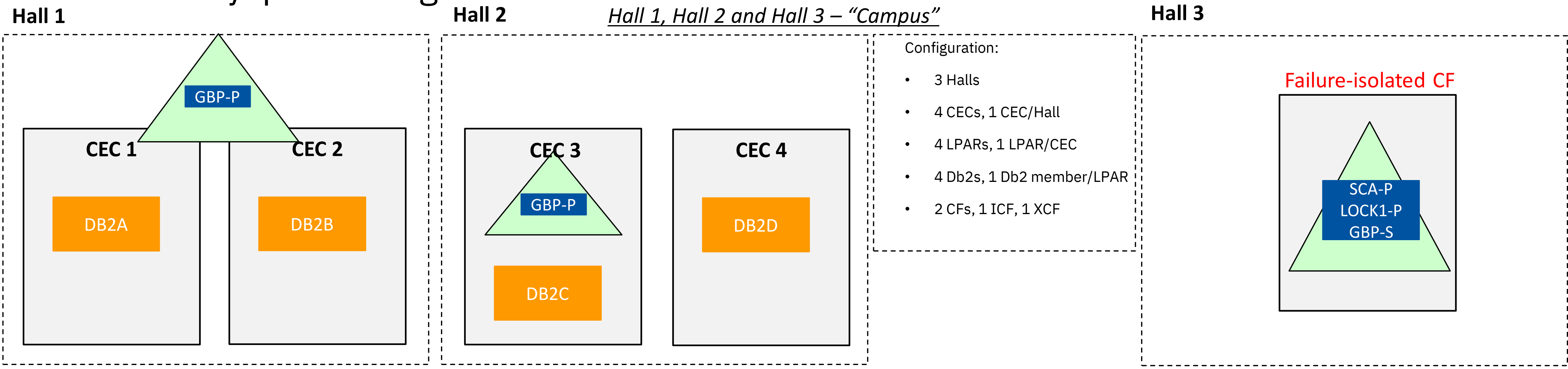
Configuration:

- 2 Halls
- 2 CECs, 1 CEC/Hall
- 4 LPARs, 2 LPAR/CEC
- 4, Db2s, 1 Db2 member/LPAR
- 2 CFs, 1 ICF, 1 XCF

- **Does this Parallel Sysplex configuration supply the continuous availability requirements ?**
  - True "Campus" continuous availability - 4-way active data sharing across 4 LPARs on two CECs (boxes)
  - A planned or unplanned CEC, LPAR or Db2 outage
    - LPAR/Db2 member failure
      - Surviving 3 LPARs and members will need to absorb 25% of the failed workload
    - CEC failure
      - Surviving/available infrastructure will need to absorb 100%
      - Planned or unplanned CEC outage will result in the surviving/available infrastructure being a single point of failure
  - **Does not provide continuous availability across a Hall failure**
    - Failure of Hall 1 will result in group wide outage
      - Db2 LOCK1 and SCA structures do not reside in failure isolated Coupling Facilities and are not being duplexed or in an isolated Hall

Achieving Continuous Availability …

- Parallel Sysplex configuration



Configuration:

- 3 Halls
- 4 CECs, 1 CEC/Hall
- 4 LPARs, 1 LPAR/CEC
- 4 Db2s, 1 Db2 member/LPAR
- 2 CFs, 1 ICF, 1 XCF

– True continuous availability recommended infrastructure - 4-way active data sharing across 4 LPARs on two CECs (boxes)

– A planned or unplanned CEC, LPAR or Db2 outage
  - CEC/LPAR/Db2 member failure
    – Surviving 3 LPARs and members will need to absorb 25% of the failed workload
    – Eliminates Hall single points of failure during a planned or unplanned outage (additional isolated Hall is expensive)

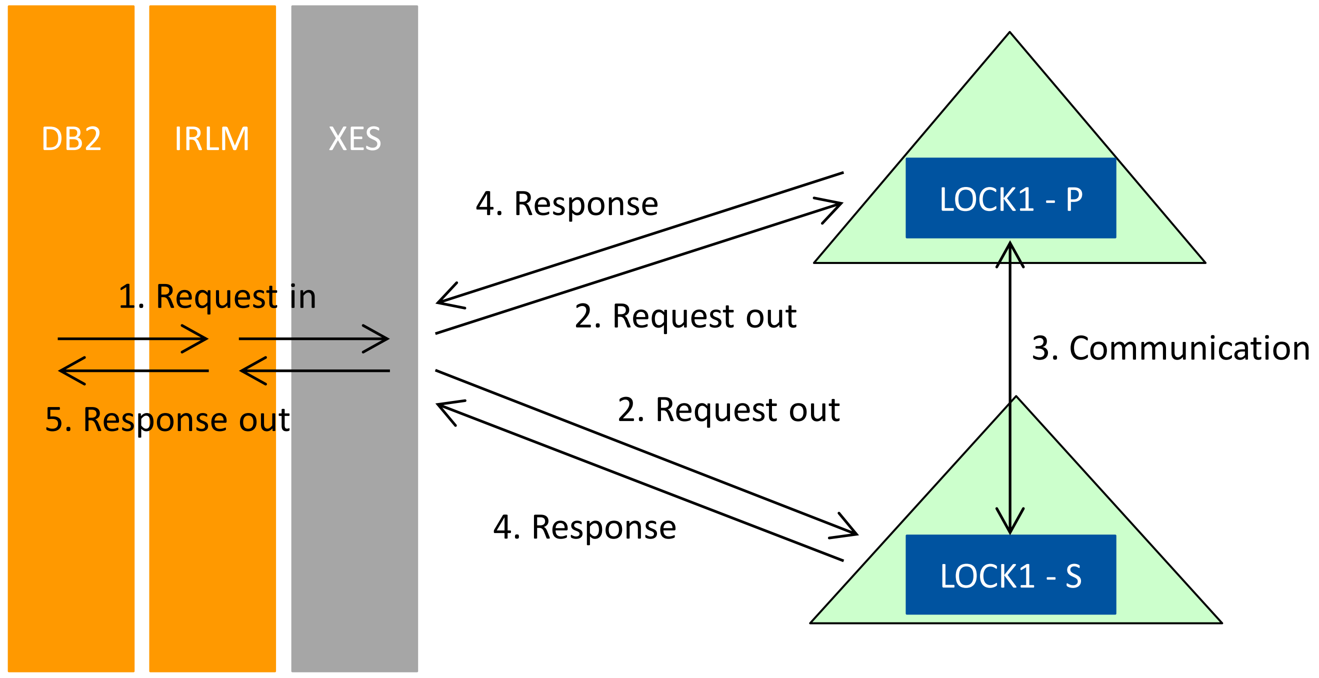
– Eliminates a Hall as a single point of failure
  - Continuous availability is achievable during any Hall unplanned outage
    – Db2 LOCK1 and SCA structures reside in a failure isolated Coupling Facility

– Alternatively considered Coupling Facility structure duplexing

Potentially a financially costly solution requiring a third isolated Hall

Db2 Synchronous CF lock duplexing

- Synchronous CF lock structure duplexing – how it works today

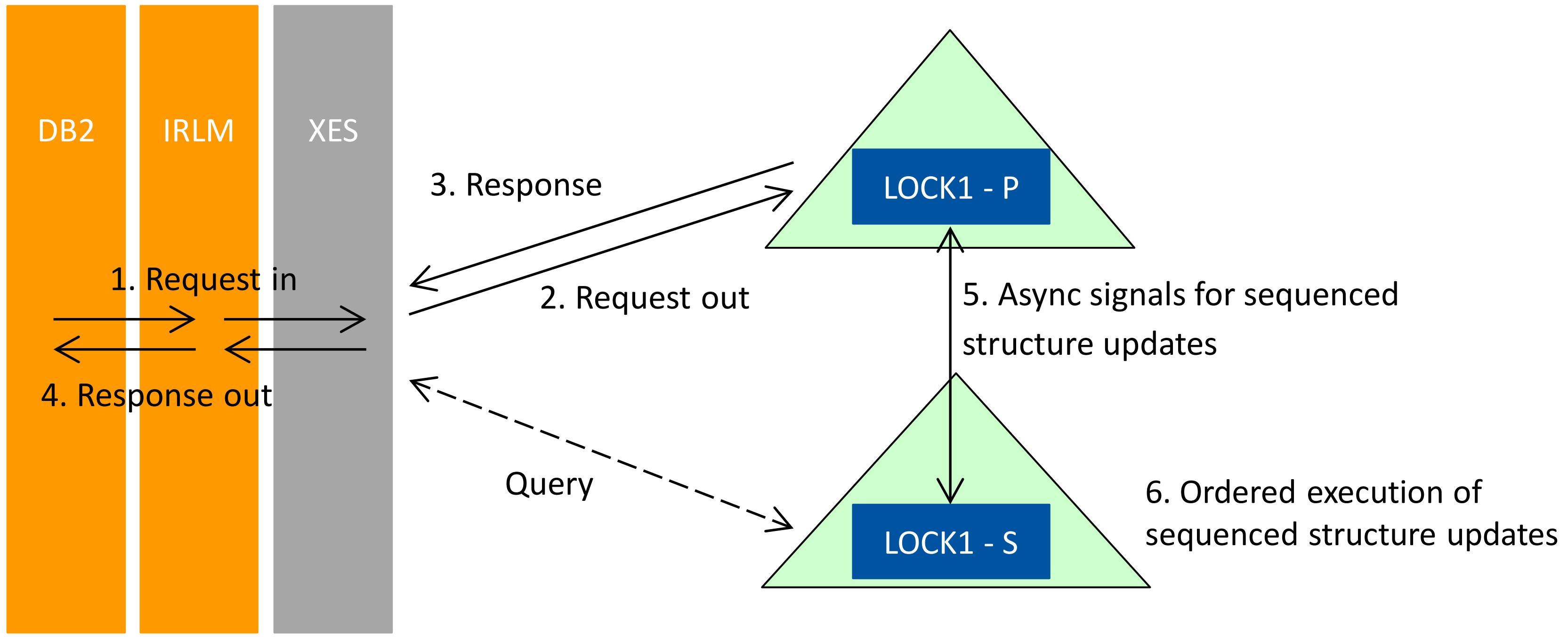


Db2 Asynchronous CF lock duplexing

- Introduced in Db2 12
  - Reduces overhead for system managed duplexing of CF LOCK1 structure
  - Secondary structure updates are performed asynchronously with respect to primary updates
  - Db2 will sync up with z/OS to ensure data integrity i.e., all modify locks have been "hardened" in the secondary lock structure before the corresponding undo/redo record for the update is written to the Db2 the active log on DASD
  - The physical log writer performs the 'sync' call to query the secondary, and it happens whenever log records get physically written to DASD, which can be earlier than commit
- Increases the practical distance for multi-site sysplex operations whilst duplexing of CF LOCK1 structure

Db2 Asynchronous CF lock duplexing …

- Asynchronous CF lock structure duplexing – how it will now work

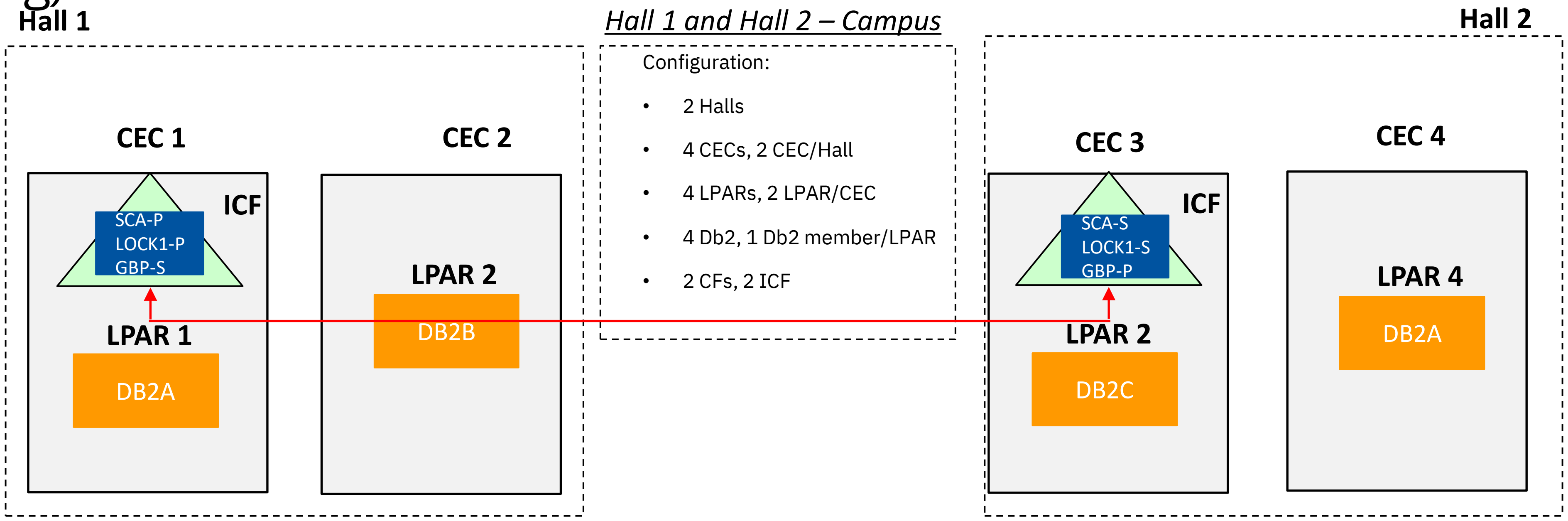

DB2
IRLM
XES
1. Request in
4. Response out
3. Response
2. Request out
Query
LOCK1 - P
LOCK1 - S
5. Async signals for sequenced structure updates
6. Ordered execution of sequenced structure updates

Db2 Asynchronous CF lock duplexing …

- Benefits
  - Cost of lock structure duplexing is significantly lower
    - Host CPU for lock requests decreases
    - IRLMs receive responses sooner
  - Existing sites using synchronous SMD should see lower host CPU cost and better elapsed times
  - More environments can now achieve higher availability in all-ICF configurations
    - Reduce risk with asynchronous SMD with less cost all round
      - Hardware maintenance
      - Capital cost for extra frames
  - Processor technology refresh applies to both host GCP and ICF engines
- But it is not free for simplex users
  - Will have to acquire ICF engines and coupling links for CF-to-CF connectivity
  - CF utilization is significantly higher for asynchronous System-Managed Structure Duplexing relative to simplex case, but it is much less than sync SMD
    - Expected to be higher than simplex because there is simply more work for the CF to do

One-Site Parallel Sysplex Configuration Options ...

- Parallel Sysplex configurations – Scenario 6 (Asynchronous CF lock structure duplexing)



Configuration:

- 2 Halls
- 4 CECs, 2 CEC/Hall
- 4 LPARs, 2 LPAR/CEC
- 4 Db2, 1 Db2 member/LPAR
- 2 CFs, 2 ICF

– True continuous availability recommended infrastructure - 4-way active data sharing across 4 LPARs on two CECs (boxes)

– A planned or unplanned CEC, LPAR or Db2 outage
  - CEC/LPAR/Db2 member failure
    – Surviving 3 LPARs and members will need to absorb 25% of the failed workload
    – Eliminates Hall single points of failure during a planned or unplanned outage

– Eliminates the need for a failure isolated (external Coupling Facility)
  - Continuous availability is achievable during any Hall unplanned outage (duplex)

Asynchronous CF LOCK1 structure duplexing significantly reduces the performance overhead (not free)
- ~1us/request
- Additional CF CPU

HiperDispatch

HiperDispatch and efficiency differences between VH,VM,VLs

- HiperDispatch (HD) plays a vital role on the latest CEC models where cache performance has such a big impact
  - PR/SM and z/OS Dispatcher interfaces are establishing an affinity between
    - Units of work and logical CPs
    - Logical CPs and physical CPs
- Impact is the increased likelihood of a unit of work being re-dispatched to the same logical CP and executing on the same or nearby physical CP
  - Optimizes the effectiveness of processor cache at every level, by reducing the frequency of processor cache misses
  - By reducing the distance (into the Nest) required to fetch data

HiperDispatch and efficiency differences between VH,VM,VLs …

- With HiperDispatch active, based on LPAR weights and number of physical CPs, PR/SM will assign logical as
  - Vertical High (VH): 1:1 relationship with physical CP
  - Vertical Medium (VM): 50% share of CP
  - Vertical Low (VL): Low share of physical CP
    - Subject to being ”parked” when not in use
    - Few seconds to “unpark”
    - ~10 second WLM interval

HiperDispatch and efficiency differences between VH,VM,VLs …

- Customer example related to zIIP pool of processors
  - Customer added 2 additional zIIP physical processor to the CEC, but did not see much better zIIP offload
  - # of logical zIIP engines and relative LPAR weights
    - Did not align with documented best practices for HD
    - Did not match with the actual demand from respective LPARs
    - No changes were made
  - Consequences
    - Unnecessary redirect of zIIP eligible workload to the GCPs
    - GCP resource was already constrained during the peak periods
    - Loss of TCO benefit and introducing elapsed time latency for application processes
  - Recommendations for this customer situation
    - Align # of logical zIIP engines and relative LPAR weight with the actual demand for zIIP capacity demand from LPAR
      - Need more dedicated zIIP capacity for the LPAR e.g.
        - Increase weight of LPAR to ensure that at least 3 VH engines are assigned
      - Should reduce zIIP redirect to GCP and increase efficiency of zIIP dispatching

Growth in CPU time/transaction as CPU busy increases

Growth in CPU time/transaction as CPU busy increases

- Source of variation
  - CPU utilization generally reflects the amount of work flowing through a fixed hardware and software configuration
    - The higher the workload rate, the higher the utilization
  - As more workflows through a fixed configuration, the efficiency of the hardware and software is reduced
    - Smaller share of hardware resources (caches, buses) available to each work unit
    - Software manages more work units – longer queues, more contention
    - CPU time per transaction will grow
  - Magnitude of the effect is related to
    - Workload characteristics
      - Higher RNI workloads (as measured at higher utilization) see higher impact
    - Size of processor
      - Smallest N-way (say 1-4) are somewhat less sensitive

Growth in CPU time/transaction as CPU busy increases …

- ROT: Approximately 4% growth of CPU/trx for each 10% growth in CPU busy
- Two implications for capacity planning
  1. May have less headroom on CEC than you think
  2. When moving a workload, it may not fit in the new container
- Example
  - Assume a workload is running 50% busy on a 2000 MIPS CEC
    - Without factoring in utilization effect, it will be called a 1000 MIPS workload
    - In fact, it may be a 1200 MIPS workload when running at the efficiency of a 90% busy CEC
  - Caution:
    - There is not room to double this workload on the current CEC
    - If moved to a new CEC or LPAR, it will likely need 1200 MIPS container (not 1000 MIPS) to fit
  - Estimating the impact – conservative approach
    - For a change in utilization of 10% plan for the capacity effect to be
      - 3% for LOW RNI workloads, 4% for AVERAGE RNI workloads, 5% for HIGH RNI workloads

“Self Healing”

“Self Healing”

- Majority of customers have established CPU capacity design usage objectives for peak/non-peak periods
  - Unknown abnormal conditions
    - Hall/Zone, CEC, LPAR failures where surviving infrastructure must absorb the lost component(s)
    - Events that trigger CPU usage spikes
      - Market surges
      - “Black Friday”
        - Stimulus checks
        - Marketing strategies
        - Political events, presidential elections
  - Finite number of engines enabled
    - Financial cost of purchasing physical engines
    - Manage software usage costs
  - On-demand capacity is enabled to supplement capacity to support peak periods and to manage unplanned peaks or usage-based failover conditions
    - OOcOD
    - CBU
- When saturation thresholds are reached failures and/or noticeable negative impact (“pain”)
  - High CPU usage exasperated
    - Limited amount of low priority workload can be pre-empted and donate cycles to high priority work when the LPAR is very high utilization
    - No transaction prioritization within the OLTP service class
      - All transactions are treated equally

“Self Healing” …

- Manual decision to add additional capacity and in not pre-planned often takes executive approval
  - Reactionary based on monitoring and alerting
  - If abnormal conditions trigger high CPU usage must accept the consequence of not reacting fast enough
    - Enterprise, application negatively impacting production events “pain”
      - Failed transactions
      - Slow transactions
      - Additional threads/agents
- As a result of high CPU and negative impacting production event
  - Answer is help at just about any cost
    - Seek executive approval
    - Add capacity
- If the result is always to add capacity, why not automate the solution?
  - Create a “Self Healing” process
    - Document the thresholds
    - Acquire executive pre-approval
    - Automatically add capacity

Questions

Thank You