IBM z Systems

2015 TPF Users Group
z/TPFDF Status Update

Chris Filachek, TPF Development Lab

March 24, 2015

IBM

Disclaimer

Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

Agenda

- **z/TPFDF Enhancements**
  - PUT 11 Enhancements and Futures
- **z/TPFDF TPFUG Requirements Update**

z/TPFDF Enhancements

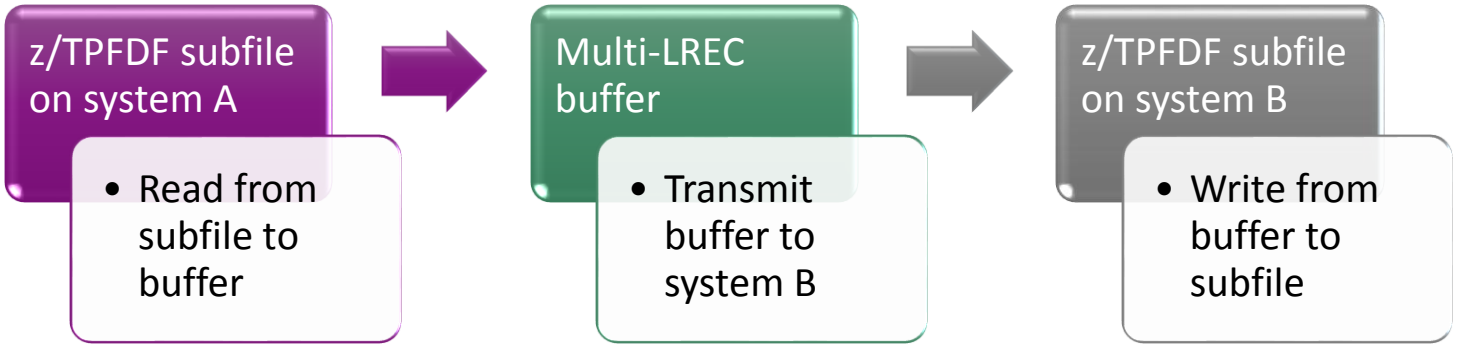
Available Enhancements (PUT 11)

- PI22341: Multiple LREC buffer direct access support
- PI24741: Prefetch prime support
- PI22609: Data collection infrastructure
- PM99272: 64-bit and baseless z/TPFDF SPMs
- PI18549: Maximum LLR size per file
- PI20336: Support for HOLD/UNHOLD APIs
- PI18980: Event message business events
- * PM98351: Timeout for ZUDFM
- * PM93257: Release records on ZUDFM INIT

* See Spring 2014 TPFUG presentation for details

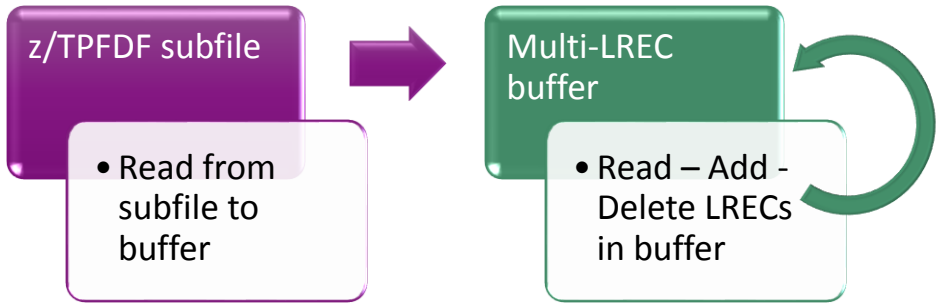
Multiple LREC Buffers on PUT 9

- Original use case
  - Read a z/TPFDF subfile into a memory buffer and transmit to another z/TPF system
    - Read buffer – read LRECs from a subfile to a buffer
    - Write buffer – write LRECs from a buffer to a subfile
- No APIs available to read or update individual LRECs in the buffer



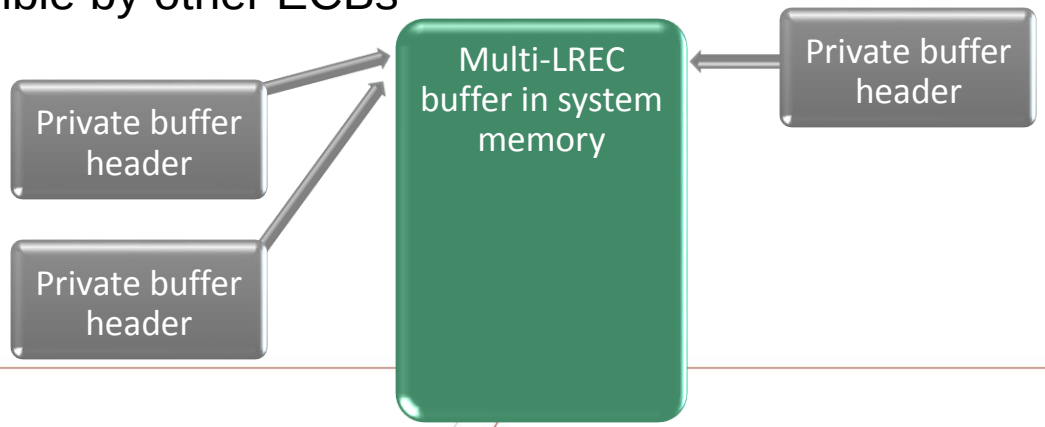
PI22341: Multiple LREC buffer direct access

- New use cases
  - Improve application performance by processing z/TPFDF subfiles in memory buffers
    - Processing is more efficient for some types of processing
    - Looping and recursive processing of LRECs that cause z/TPFDF to repeatedly move back and forth between prime & overflow blocks
  - Simplify application programming for temporary workfiles
    - Workfiles that are manipulated but not saved
    - Does not require DBDEF definition
- New multiple LREC buffer APIs to manipulate LRECS in the buffer
  - Read with and without keys
  - Add / replace / delete LRECs
  - Retain and restore position



PI22341: Sharing Multiple LREC buffers

- New use case
  - Store read-only z/TPFDF subfiles in system memory for shared access by multiple ECBs
    - Multiple ECBs can read LRECs from the same shared buffer
    - Each ECB has a private buffer header to access to the shared buffer
    - Buffer header manages private ECB information
      - Current LREC position
      - Search keys
- New multiple LREC buffer APIs to publish and access shared buffers
  - Publish a buffer so it is accessible by other ECBs
  - Get / release buffer header
  - Copy multiple LREC buffers



Before Prefetch Prime: Sequential Reads

- A subfile is opened: DBOPN
  - The SW00SR is setup
  - No records are read for the subfile
- A subfile is accessed for the first time: DBRED / DBADD / etc.
  - z/TPFDF reads the prime block from disk (FINDC)
  - z/TPFDF waits for the read to complete (WAITC)
- If multiple subfiles are accessed by an ECB, all prime blocks are read sequentially!
  - FINDC–WAITC, FINDC-WAITC, FINDC-WAITC, ...
- Application response times may be limited by sequential read process

PI24741: Prefetch prime support

- Reduce application latency by reading the prime blocks for multiple z/TPFDF subfiles in parallel
  - Open a subfile (DBOPN) with the PREFETCH PRIME parameter
    - The SW00SR is set up
    - z/TPFDF ***reads the prime block*** from disk (FINDC)
    - z/TPFDF ***does not wait*** and returns to the application
  - Open additional subfiles with the PREFETCH PRIME parameter
  - Finish by issuing the z/TPFDF DBWAIT / dfwait() API
    - Wait for all I/O to complete
    - Complete setting up all SW00SRs that used PREFETCH
  - Access subfiles normally using existing z/TPFDF APIs
- Supports HOLD and NOHOLD options
  - If opening with HOLD, DBOPN may issue a WAITC macro to avoid deadlock conditions

Before Data Collection Infrastructure

- z/TPFDF data collection - ZUDFC command
  - Unique data collection separate from z/TPF Data Collection (DC) and Continuous Data Collection (CDC)
  - z/TPFDF data is not shared with z/TPF DC or CDC
  - ZUDFC usually run at a different time than DC
  - Difficult to analyze system performance over a given time interval
- Measurable overhead while ZUDFC collection was on
  - Shared collection area updated for every count event
  - ZUDFC turned on only for short periods of time

PI22609: Data Collection Infrastructure

- New z/TPFDF data collection method
  - Same type of counts as ZUDFC
  - Counts are accumulated in the SW00SR
  - Added to a shared collection area in memory when subfile is closed
    - Used by existing ZUDFC commands
  - z/TPFDF counts are collected continuously
    - Collection method is more efficient, allowing it to be always on
- New z/TPFDF user exit during close processing
  - Applications can query counts for the subfile
  - Example: Real-time monitoring of subfile size (number of overflow records read) to detect anomalies as early as possible
- Prepares z/TPFDF data collection for integration into z/TPF data collection and CDC....

Future Enhancement – Collection Integration!

- z/TPF data collection
  - z/TPFDF data collected by z/TPF data collection
  - z/TPFDF reports produced by new data reduction tool
    - Included in staged rewrite of data reduction tool
    - New data reduction tool planned for Linux
- Continuous Data Collection (CDC) and Tivoli Monitoring
  - Display rates and counts for TOP “n” z/TPFDF files

TPFDF Rates Based on TOPn DBRED

SS Name	File ID	DBRED per Sec	DBADD per Sec	DBADR per Sec	DBCKP per Sec	DBCLS per Sec	DBCPY per Sec	DBCRE per Sec	DBDEL per Sec	DBD per S
BSS	B214	74.0	0.0	0.0	0.0	9.8	0.0	0.0	41.1	1.6
BSS	B211	14.8	0.0	0.0	0.0	0.0	0.0	0.0	1.6	0.0
BSS	B212	8.2	0.0	0.0	0.0	1.6	0.0	0.0	90.5	1.6
BSS	B227	6.9	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0

Future Enhancement - z/TPFDF Cache

- Today
  - Files are always read from disk or VFA
    - Disk or VFA I/O required to read every prime and overflow record
  - Requires VFA sync if loosely coupled (VFA sync locks)
- Tomorrow
  - Option to cache subfiles to reduce I/O and improve performance
    - Store copy of full subfile in logical record cache (local memory)
  - Use cache copy when opening read-only
    - Copy full subfile from cache to private ECB memory in SW00SR
      - Avoids DASD and VFA I/O and doesn't require VFA sync locks
    - Supports all existing z/TPFDF APIs with no application changes
  - Use disk copy when opening read-write
    - Invalidate cache entries when subfile is updated
    - Uses coupling facility (CF) when z/TPF is loosely coupled
  - Best for "mostly read-only" files and where most records in chain are read

Future Enhancement – CRUISE Restore

- Performance improvements for CRUISE Restore processing
  - New use case – copying production databases to test systems
    - Many CRUISE copy and restore sessions to copy subset of databases
    - Long restore runtime makes process time consuming
  - Making CRUISE Restore more efficient...
    - Change single ECB restore processes to multi-ECB
    - Improve multi-ECB load balancing during tape roll-in
      - Current options cause excessive DEFRC processing or is difficult to tune
    - Option to skip intermediate record copies and restore directly to fixed records

PM99272: 64-bit and baseless z/TPFDF SPMs

- Previously
  - z/TPFDF Structured Programming Macros (SPMs) only allowed with:
    - 31-bit addressing mode and R8 as the only base register
- Now
  - z/TPFDF SPMs can now be used with:
    - 31-bit or 64-bit addressing mode
      - Easily reference 64-bit memory areas
      - Allows assembler programs to interface more easily with C/C++ programs
    - Any valid base register or no base register (baseless)
      - Greater flexibility in register selection
      - Expand programs without needing to split them into multiple modules

PI18549: Set maximum LLR size per file

- Previously:
  - One maximum LLR size set for all z/TPFDF files
    - #LLRMLR equate in ACPDBE macro
  - Prevents inadvertently storing large amounts of data in a single LLR
  - Value had to be set based on the largest maximum of any file
- Now:
  - Different maximum LLR size can be set for each z/TPFDF file
    - MAXLLR parameter on DBDEF macro
  - If specified, DBDEF setting is used (MAXLLR) instead of system setting (#LLRMLR)
  - Provides more flexibility in enforcing data integrity

PI20336: Support HOLD/UNHOLD APIs

- Previously
  - Application determines HOLD/NOHOLD when subfile opened
    - Must close and re-open the subfile if you need to change the hold status
    - Pessimistic locking
      - Lock subfile if there is a chance it will be updated
      - Subfiles are locked even if no updates are done
- Now
  - Hold subfile (DBHOLD / dfhold)
    - Hold subfile after it has been opened and read by the application
    - May require application re-process subfile if sequence counter has changed or not available
  - Unhold subfile (DBUHLD / dfuhld)
    - Choose to file or discard changes
    - Unhold subfile but keep open read-only

PI18980: Event message business events

- Previously:
  - Business events only included support for signal events
- Now:
  - Business events supports data events for z/TPFDF files
- Co-req z/TPF APAR is PJ42280

- See “Event Message Business Events” presentation for details!

z/TPFDF TPFUG Requirements Update

Requirements with Changed Status

Requirement	Description	Was	Now
DF12198	Multi-LREC Buffer Allow Record Access	Accepted	Available (PI22341)
DF13200	Process Files from Heap	Accepted	Available (PI22341)
DF14202	CRUISE Restore Improvement	New	Accepted
DF00165	Integrate TPFDF Data Collection and reporting	Likely	Accepted
DF05182F	Export/Import TPFDF LRECs to/from XML Format	Likely	Accepted
DF00153	C++ APIs	Accepted	Likely

Trademarks

- IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.
- *(Include any special attribution statements as required – see Trademark guidelines on https://w3-03.ibm.com/chq/legal/lis.nsf/lawdoc/5A84050DEC58FE31852576850074BB32?OpenDocument#Developing%20the%20Special%20Non-IBM%20Tr)*

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.
- 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.
- This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.