#IDUGDb2

IDUG Db2 Tech Conference, Rotterdam, Netherlands 2019

Continuous Delivery with Db2 12 for z/OS Best Practices

John Campbell

IBM Db2 for z/OS Development

Platform: Db2 for z/OS

IDUG

Leading the Db2 User Community since 1988

Old Db2 for z/OS Strategy for delivering new function

- We deliver most of our new function in a new release ~every 3 years
- Db2 for z/OS is on 3 year cycles, but many of our customers are on 4 year cycles, hence the interest in skip release migrations
- We develop or retrofit a very limited number of new features in the service stream, but only if urgent and generally low risk
- Deployment of new releases is seen as a disruption by our customers
- Many of our customers want new features delivered much faster
- Industry and customer trend is to move away from monolithic code delivery towards continuous delivery model
- IBM is moving towards continuous delivery model
- Time for us in Db2 for z/OS to change

New Db2 for z/OS Strategy for delivering new function

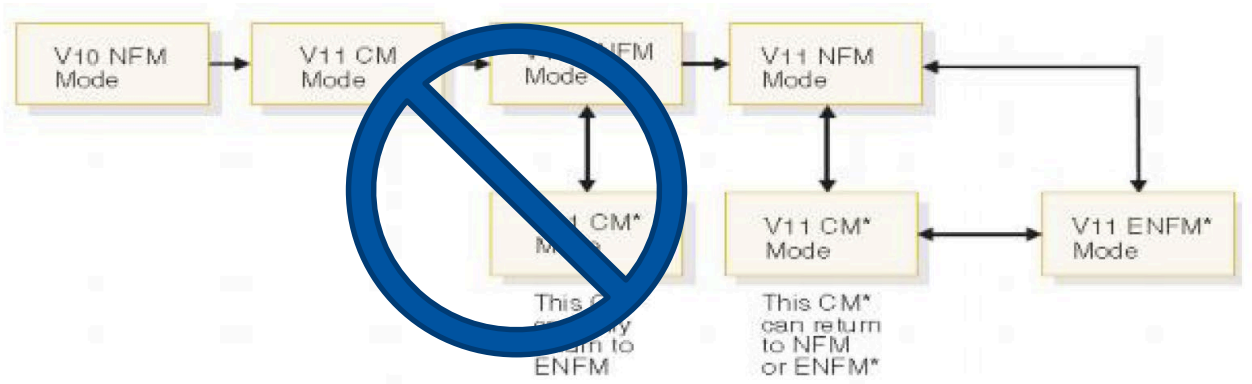
- **We are dedicating ourselves to going forward on a continuous delivery model**
  - Radical internal changes are required within Db2 for z/OS Development to do this
- **Db2 12 for z/OS is the starting point after GA**
  - There will be significantly higher volume of continuously delivered items
- **Customers will see a single maintenance stream for Db2 12 for z/OS, with the new function delivered into that**
  - The function will be designed to be easily consumable
- **Point releases or versions will be a very rare exception**
  - There are reasons why we might want to have a point release or new version
  - e.g., adopt a new compiler, extend control structures, enable an architecture level set
- **Db2 for z/OS Development will have relentless focus on maintaining continuous production level reliability for you in the service stream**
- **We are dedicated to doing this**
  - We will control the input to “the pipe”, the size and risk of the items
  - Increased internal focus on function and performance regression testing
  - We will deliver new function when the quality is right, and not based on a promised date for delivery

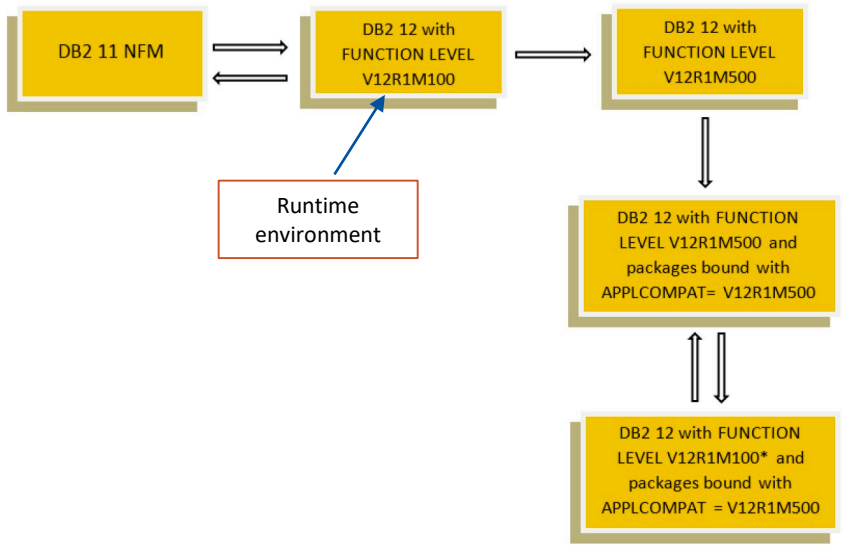
Understand Continuous Delivery starting with Db2 12 for z/OS

- **With Continuous Delivery, there is a single delivery mechanism for defect fixes and enhancements**
  - PTFs (and collections of PTFs like PUTLEVEL and RSU) → same as today
- **With Continuous Delivery, there are four Db2 for z/OS levels**
  - Maintenance level (ML) = code level – lifted by applying maintenance
    - Also known as code level - contains defect and new enhancement fixes
    - Most new functions are shipped disabled until the appropriate new function level is activated
  - Catalog level (CL) - vehicle to enable new FL - accumulative (skip level possible)
    - Db2 Catalog changes that are needed for some FLs
  - Function level (FL) – needs to be activated - accumulative (skip level possible)
    - Introduces new Db2 for z/OS features and functionality
    - No impact or change in existing application behaviour
  - APPLCOMPAT level (AC) – set by application - provides an “island of stability” for a given application
    - Determines SQL function level of applications – can increase FL of the application (and fallback)
    - AC must be advanced to exploit new SQL function
    - AC level in BIND/REBIND of package must be <= FL and rules over FL
    - Freezes new SQL syntax even if FL is later moved back to earlier level
- **Minimum starting point for Continuous Delivery is Db2 12 for z/OS FL=V12R1M500**

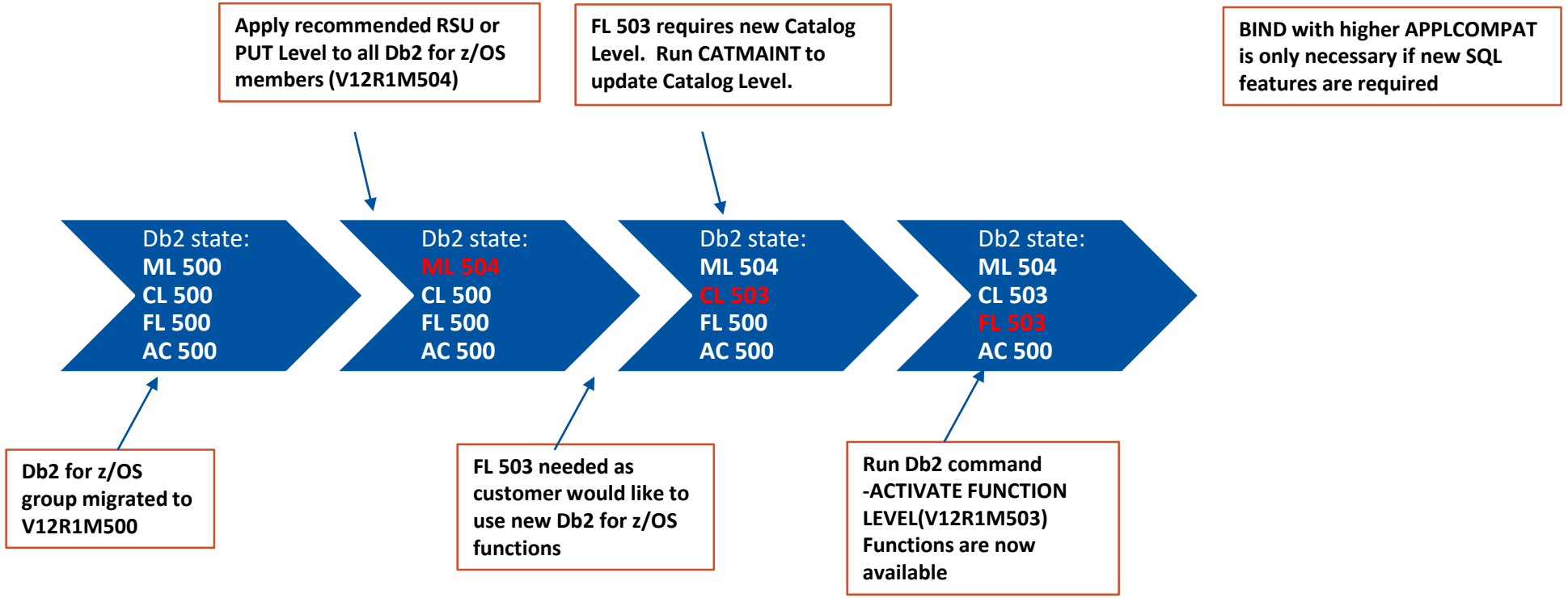
Understanding new function levels

- **CM / ENFM / NFM no longer used**
- **Function Level V12R1M100**
  - Similar to CM / BNFA
  - Db2 12 for z/OS engine and catalog / directory
  - DSNTIJTC (CATMAINT) to get there
  - Fallback to Db2 11 for z/OS NFM possible
- **Function Level V12RM15nn**
  - Similar to NFM /ANFA
  - New functionality available
  - Command –ACTIVATE FUNCTION LEVEL(V12R1M5nn) to get there
  - Fallback to Db2 11 for z/OS NFM no longer possible (PIT recovery would be required)





Example of how to get to a new function level



Change in strategy for APPLCOMPAT

- **No need to force the rebind all packages with a new, higher APPLCOMPAT level**
- **APPLCOMPAT will now have many more versions to support many Function Levels**
- **Must still rebind a package with a higher APPLCOMPAT level in order to exploit new SQL DML, SQL DDL, SQL DCL, and XML function**
  - Applications can only use new SQL if the packages are bound with the necessary and required Application Compatibility (APPLCOMPAT)
    - Packages can only be bound with an APPLCOMPAT less or equal to the current FL

- **Still recommended best practice to regularly rebind all packages to**
  - Benefit from latest run time performance improvements
  - Gain exposure to new access path selection improvements
  - Benefit from defect fixes
  - Reduce exposure to latent issues seeded previously

Is APPLCOMPAT a 'sticky' option on BIND/REBIND?

- BIND REPLACE does **not** reuse any bind option from the existing package if the option is not explicitly specified
- SQL statements can be totally different so BIND REPLACE is considered a new bind
- REBIND and BIND COPY are the only subcommands that reuse the existing/source package's option
- This is true in all Db2 for z/OS releases and not just Db2 12 for z/OS

Setting CURRENT APPLICATION COMPATIBILITY special register

- **Db2 11 for z/OS**
  - Value can be >= APPLCOMPAT level of the executing package but not > current zparm APPLCOMPAT
- **Db2 12 for z/OS**
  - Value has to be <= APPLCOMPAT level of the executing package, independent of the current Db2 Function Level
  - Why?
    - To be able to control application use of new functionality
    - Avoid breaking applications when activating a lower Function Level i.e., fallback

Preventative service planning for Db2 12 for z/OS and Continuous Delivery

- **Apply preventative maintenance every 3 months based on rolling calendar**
  - Use RSU instead of PUT to be less aggressive on applying non-HIPER maintenance
  - 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
      - Ensure that RSU-only service is installed by adding the SOURCEID (RSU*) option in the supplied APPLY and ACCEPT jobs
    - In addition, apply two “minor” packages in between covering HIPERs, PE resolution fixes and forward fit of blocked maintenance
- **Continuous program to review Enhanced HOLDDATA on a weekly basis**
  - When rolling out maintenance package to production and after post production cutover
  - For vicious HIPER problem where no operational bypass/workaround, expedite fix into production after 2 weeks in test
  - Others can be deferred until the rollout of the next maintenance drop
    - By applying operational bypass/workaround in production
    - Only “low grade” problem in production
    - Not applicable

Function Level Adoption – Best Practices 1/2

- **Prioritise maintaining production stability over new function adoption**
- **If applying preventative maintenance only once or twice per year, then apply more frequently**
  - Every 3 months using a rolling calendar
- **PTFs (RSUs…) are applied that may increase the Maintenance Level (ML) of a Db2 for z/OS subsystem**
- **After system is stable on new maintenance level (ML) – may be after a complete maintenance cycle**
  - Execute (If Any) Catmaint
    - **After execution of Catmaint, the subsystem can only be started with a ML that supports the catalog**
  - Activate Function Level (FL)
    - **After activating a new FL, the subsystem can only be started with a ML that supports the FL**
    - New function not related to SQL DML, DDL and DCL syntax is available
    - REBIND of packages with any APPLCOMPAT would pick up optimizer enhancements
    - Non-stabilized dynamic SQL would pick up optimizer / other non-APPLCOMPAT related enhancements

Function Level Adoption – Best Practices 2/2

- **After Function Level is considered stable - allow new application feature rollout**
  - REBIND DBA packages to allow new DDL to be utilized
  - REBIND application static packages with higher APPLCOMPAT to exploit SQL DDL/DML new functions/behaviors
  - REBIND dynamic packages with higher APPLCOMPAT to allow new SQL functions to be used
  - REBIND distributed packages (***in separate collection) with higher APPLCOMPAT to allow new SQL functions to be used
    - Switch applications to use new distributed package collection
  - Leverage PLANMGMT extended
    - Use REBIND SWITCH (PREVIOUS) to restore static SQL packages to prior runtime structures
    - Use REBIND SWITCH (PREVIOUS) for dynamic SQL packages would restore prior APPLCOMPAT
    - ***switching to prior collid for distributed dynamic would restore APPLCOMPAT

Db2 12 for z/OS Migration – Quick Hits

- **Db2 Connect – Situation until recently prior to applying PTF for APAR PH08482**
  - Any level of Db2 Connect drivers should work with Db2 12 for z/OS, both before and after new function is activated with no behavior change – level should still be in-service
  - Data server clients and drivers must be at the following minimum levels to exploit Db2 for z/OS function-level application compatibility (APPLCOMPAT) of V12R1M**501** or greater:
    - IBM® Data Server Driver for JDBC and SQLJ: Versions 3.72 and 4.22, or later
    - Other IBM data server clients and drivers: Db2 for Linux, UNIX, and Windows Version 11.1 Modification 2 Fix Pack 2, or later
  - New ClientApplCompat (ODBC) and clientApplcompat (JDBC) property setting allows you to control the capability of the client when updated drivers ship changes to enable new server capability
    - You might want specific control of driver capability when:
      - Db2 client driver introduces new behavior currently not controlled by Db2 application compatibility
      - Change needs to be controlled at the application level to ensure compatibility with new behavior
  - ClientApplCompat/clientApplcompat setting of V12R1M500 is required to access Db2 12 for z/OS Server capability shipped after GA at function levels beyond Db2 12 for z/OS FL=V12R1M**500**
  - Db2 Connect Server gateway does NOT support the new ClientApplCompat/clientApplcompat property

Db2 12 for z/OS Migration – Quick Hits …

- **Db2 Connect – new behavior after applying PTF for APAR PH08482**
  - Makes setting of ClientApplCompat/clientApplcompat property optional
    - Customer no longer forced to have the setting
  - No changes are required in Db2 Connect level and configuration
  - All customers must upgrade to at least Db2 Connect V11.1 M1 FP1 or higher in order to run DRDA applications where APPLCOMPAT > V12R1M500 i.e., exploit new function delivered with Continuous Delivery
  - Db2 Connect Server gateways will also need to be upgraded to at least Db2 Connect V11.1 M1 FP1 to access packages using APPLCOMPAT > V12R1M500
  - When ClientApplCompat/clientApplcompat is set, Db2 will perform validation checking where there are changes in DRDA message flows i.e., check the underlying infrastructure and avoid application incompatibilities

Db2 12 for z/OS Migration – Quick Hits …

- **Db2 Connect – new behavior after applying PTF for APAR PH08482 …**
  - Going out into the future when not setting ClientApplCompat/clientApplcompat there are consequences in terms of risk when DRDA flows changes as existing applications
    - So far in Db2 12 there are no changes in DRDA flows, no changes are in plan, but at some point it will likely happen
    - Applications may break in the future when DRDA transactions run with package where APPLCOMPAT > FL500
    - If an application breaks then Db2 will not provide server support to allow these broken applications to run i.e., no more new DDF_COMPATABILITY zparm settings
  - So what are the your options
    1. Rebind driver packages in the NULLID collection and back level the APPLCOMPAT setting
       This is a "one size fits all" solution to fallback to an earlier APPLCOMPAT
    2. “Penalty box” the problem applications
       a. Switch the problem applications out to use the driver packages in a different collection which has a back levelled APPLCOMPAT setting, or
       b. Switch all the good applications out into a new collection using driver packages with the new APPLCOMPAT setting and leave the problem applications still using the driver packages in the NULLID or different collection but with the driver packages running a back levelled APPLCOMPAT setting

Db2 12 for z/OS Migration – Quick Hits …

- **Db2 Connect – new behavior after applying PTF for APAR PH08482 …**
  - General best practice recommendation
    - When migrating to Db2 12 - all DRDA applications should continue to use the driver packages in the NULLID collection
    - These packages can have an APPLCOMPAT setting of V10R1, V11R1, V12R1M100 or V12R1M500 depending on where you are in the migration process
    - The APPLCOMPAT setting for the driver packages in the NULLID collection should not advance beyond V12R1M500
    - When specific applications and their application servers want to use new function requiring APPLCOMPAT setting > V12R1M500, these application servers should switch away from using the driver packages in the NULLID collection to a new collection (e.g., V12R1M503) where the driver packages are bound with a higher APPLCOMPAT setting (e.g., driver packages bound with APPLCOMPAT V12R1M503 in collection V12R1M503)

Please fill out your session evaluation before leaving!

John Campbell

IBM Db2 for z/OS Development

campbelj@uk.ibm.com

IDUG

Leading the Db2 User Community since 1988