BROADCOM

Setting Up Self-Service Parallel Development with Endevor Dynamic Environments

Vaughn Marshall
Endevor Product Manager

Parallel Development is a Key Agile Practice

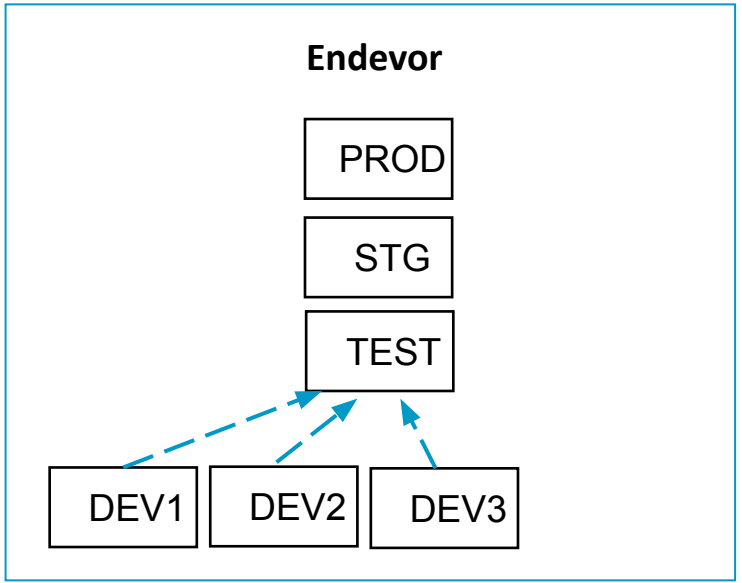
- A key practice for developer agility is the ability to work on changes **in isolation** from other changes
- Testing is easier if changes are not co-mingled
- Avoids creating unnecessary dependencies
- Can choose when and where to integrate changes (good practice is to merge trunk changes into parallel stream “Early and Often”
- Promote when ready at any time

Parallel Development In Endevor

- There are now more options than ever for Parallel or Concurrent development in Endevor:
  - Dynamic Environments ***NEW***
  - Sandboxes
  - Bridge for Git
  - Eclipse workspaces
  - Zowe workspaces
- Merging your changes with others can be done via a number of options (depending on where you are working):
  - PDM
  - Git merge
  - Eclipse merge
  - VS Code merge
  - Zowe workspace merge

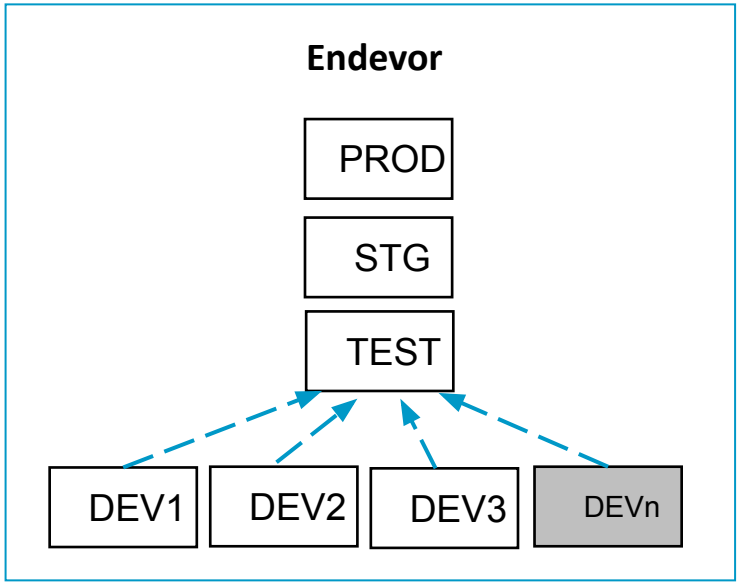
Taking Inspiration from the Past

- The need for parallel working areas is not new
- In the past, administrators configured parallel paths to production, with a merge point “up the map”
- Parallel environments were limited
  - Sometimes associated with entire releases
- Unlimited ability to branch in distributed tools led to best practice of making a parallel feature branch **for every change**



Dynamic Environments

- Dynamic Environments is a new feature that allow admins to facilitate **self-service of brand new parallel environments**
  - Addresses some key issues with Sandboxes
  - Can use any system/subsystem in the parallel environment
  - Environments can be used with any interface of choice
- Two key features enable this:
  - **Deferred File Creation** – files are only allocated when used, so the usual space overhead of a full environment is avoided
  - **New SCL** to minimize the administration required to define an environment



Deferred File Creation

- Deferred File Creation allows administrators to set up datasets that will only be allocated on first use:
  - Base, Delta, and Source Output libraries
  - Processor Output libraries (OBJLIB, LOADLIB, LISTLIB, and so on)
  - Supports PDS, PDSEs, Sequential files, ELIBs, USS Files
  - Allows Post Create Initialization
- Used to ensure unnecessary disk is not used for parallel development workspaces such as Sandboxes or Dynamic Environments
- Type definition files (Base, Delta, Source Output) must be defined with symbols in their names
  - Explicit files in Type Definitions must exist
- Files that are processor inputs should use ALLOC=COND
  - Input files will be skipped if ALLOC=LMAP or ALLOC=PMAP is used

Enabling Dynamic Environments and Deferred File Creation

- Deferred File Creation
  - PTFs SO12441 and SO13082 must be applied
    - USS Path Names (PTFs LU02424 and LU02425)
    - Initialization (PTFs LU01495, LU01496, and LU01497)
    - ELIB Support (PTFs SO15778 Endevor Base & SO15779 Japanese)
  - Enable via C1DEFLTS table
  - Specify the Deferred File Creation member of the PARMLIB data set using the DFCMBR= XXXX
- Dynamic Environments
  - PTF SO14939 must be applied
    - SO14940 (Japanese Localization)
  - MCFs defined at creation time
  - All other libraries can be allocated on first use with Deferred File Creation
  - Global Control File has to be allocated. Sample JCL shown to right.
  - A new TYPE=MAIN parameter GCFDSN specifying the GCF file needs to be added to C1DEFLTS

```
C1DEFLTS TYPE=MAIN,
GCFDSN=IPRFX.IQUAL.GCF, X
```

```
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD  *
  DEFINE CLUSTER                          -
    (NAME('IPRFX.IQUAL.GCF')              -
     INDEXED                              -
     KEYS(64 8)                           -
     RECORDSIZE(640 3070)                 -
     FREESPACE(30 30)                     -
     SHAREOPTIONS(3 3)                    -
     SPEED                                -
     BUFFERSPACE(18944)                   -
     STORAGECLASS(NDVRPOOL)               -
     DATACLASS(DEFAULT))                  -
  DATA                                    -
    (NAME('IPRFX.IQUAL.GCF.DATA') -
     VOLUMES(VVOLSER)                     -
     CYLINDERS(2 1)                       -
     CONTROLINTERVALSIZE(8192))           -
  INDEX                                   -
    (NAME('IPRFX.IQUAL.GCF.INDEX') -
     VOLUMES(VVOLSER)                     -
     TRACKS(1 1)                          -
     CONTROLINTERVALSIZE(2560))
/*
```

Deferred File Creation - DFCMBR Examples

Documentation

Example 1: Processor output library entries using either LIKE statement or explicit attributes

```
  DSN
    '&C1EN/&C1S#/&C1SY/&C1SU/&C1ELEMENT/&C1TY'
    .
  DSN
    'Stage ID: &C1SI   Stage Name: &C1ST'
    .
*
* Processor output libraries.
*
  DSN 'NDV.&C1SY..&C1SU..&C1ST..ELMOUT'
    LIKE 'NDV.&C1SY..GA.&C1EN(1,1)&C1S#..ELMOUT'
    .
  DSN 'NDV.&C1SY..&C1SU..&C1ST..LOADLIB1'
    UNIT SYSDA
    RECFM U
    BLKSIZE 32760
    SPACE CYLS
    PRIMARY 1
    SECONDARY 5
    DSNTYPE PDSE
    DIRBLKS 91
    .
  DSN 'NDV.&C1SY..&C1SU..&C1ST..LOADLIB2'
    LIKE 'NDV.&C1SY..GA.&C1EN(1,1)&C1S#..LOADLIB2'
    BLKSIZE 32760
```

Example 2: How good naming convention allows simplification of DFCMBR

```
DSN NDV.NDVR181.&C1EN&C1ST..COPYLIB'
LIKE NDV.NDVR181.PRDPROD.COPYLIB'
.
DSN NDV.NDVR181.&C1EN&C1ST..SRCLIB'
LIKE NDV.NDVR181.PRDPROD.SRCLIB'
.
DSN NDV.NDVR181.&C1EN&C1ST..PLIPGM'
LIKE NDV.NDVR181.PRDPROD.PLIPGM'
.
DSN NDV.NDVR181.&C1EN&C1ST..&C1TY'
LIKE NDV.NDVR181.PRDPROD.&C1TY'
.
DSN NDV.NDVR181.&C1EN&C1ST..DELTA'
LIKE NDV.NDVR181.PRDPROD.DELTA'
.
DSN NDV.NDVR181.&C1EN&C1ST..LISTLIB'
LIKE NDV.NDVR181.PRDPROD.LISTLIB'
.
DSN NDV.NDVR181.&C1EN&C1ST..LOADLIB'
BLKSIZE 32760
LIKE NDV.NDVR181.PRDPROD.LOADLIB'
.
```

Example 3: How to correctly sequence DFCMBR entries

```
* Type VBDATA base data set – VB-259
* This must be before the entry with &C1TY as last qualifier
  DSN 'NDV.&C1SY..&C1SU..&C1ST..VBDATA'
    UNIT SYSDA
    RECFM VB
    LRECL 259
    BLKSIZE 27998
    SPACE CYLS
    PRIMARY 1
    SECONDARY 1
    DIRBLKS 90
    .
* All other Type base data sets – FB-80
  DSN 'NDV.&C1SY..&C1SU..&C1ST..&C1TY'
    UNIT SYSDA
    RECFM FB
    LRECL 80
    BLKSIZE 27920
    SPACE CYLS
    PRIMARY 1
    SECONDARY 1
    DIRBLKS 90
```

Letting Developers Serve Themselves

- Administrators will be best served if they aren't a bottleneck for new environment creation
- Developers can leverage types & processors as a mechanism to request / spin up a new environment
- Create a type, such as "DYNENV", and have the Generate processor run SCL to spin up the environment. Have the Delete processor clean up the environment.
  - Environment must be empty of elements to delete it - this will prevent any unexpected loss of work before the work is promoted out of the environment
- MCFs must be created before environment, the others can be done via DFC
- You can also use `zowe endevor submit scl` to spin up environments via external orchestrators like Jenkins

Dynamic Environments SCL

- Define from an existing static environment using LIKE keyword

```
DEFINE ENVIRONMENT '&C1ELEMENT'
TITLE 'DYNAMIC DEVELOPMENT ENVIRONMENT &C1ELEMENT'
STAGE ONE MCF '&MCF1'
STAGE TWO MCF '&MCF2'
LIKE &C1ENVMNT
  NEXT ENVIRONMENT &C1ENVMNT
.
```

- Clone systems, subsystems and types (selectively or all together)

```
CLONE SYSTEM *
   FROM ENVIRONMENT &C1ENVMNT
   TO ENVIRONMENT &C1ELEMENT
   INCLUDE SUBSYSTEMS
   INCLUDE TYPES
.
```

- Clean up environments

```
DELETE ENVIRONMENT '&C1ELEMENT' .
```

Sample Processor

```
//*--------------------------------------------------------------------*
//* GSANDDFC
//*         GENERATE DYNAMIC ENVIRONMENT
//*--------------------------------------------------------------------*
//GSANDTST PROC DEFINE=Y,
//         HLQ='BST.ENDTEST2',
//         MCF1='&HLQ..&C1ELEMENT..S1',
//         MCF2='&HLQ..&C1ELEMENT..S2',
//         VIO=SYSDA
```

```
//ALOC     EXEC PGM=IDCAMS
//SYSPRINT DD   DSN=&&IDMSGS,DISP=(SHR,PASS)
//SYSIN    DD   *
 DEFINE CLUSTER (NAME('&MCF1') -
   IMBED -
   SPEED -
   UNIQUE -
   FREESPACE(30 30)    -
   RECORDS(1000 1000)  -
   STORAGECLASS(TSO)   -
 /*VOLUME(NDVS03)*/ -
   RECORDSIZE(640 1200) KEYS(28 0) SHR(3 3)) -
  DATA  (NAME('&MCF1..DATA') -
   CISZ(8192)) -
  INDEX (NAME('&MCF1..INDEX') -
   CISZ(2048))
 /* DEFINE MCF AT STG2
 DEFINE CLUSTER (NAME('&MCF2') -
   IMBED -
   SPEED -
   UNIQUE -
   FREESPACE(30 30)    -
   RECORDS(1000 1000)  -
   STORAGECLASS(TSO)   -
 /*VOLUME(NDVS03)*/ -
   RECORDSIZE(640 1200) KEYS(28 0) SHR(3 3)) -
  DATA  (NAME('&MCF2..DATA') -
   CISZ(8192)) -
  INDEX (NAME('&MCF2..INDEX') -
   CISZ(2048))
//*
```

Define MCF VSAM for environment

BROADCOM

Sample Processor

Use Admin Batch Utility to clone existing environment

Clone all Systems, Subsystems & Types in new environment

```
//ADMIN    EXEC PGM=ENBE1000,ALTID=Y,
//         PARM='MSGS1(CUSTMSG1),MSGS2(CUSTMSG2),SCLIN(CUSTSCL)'
//CUSTMSG1 DD DSN=&&BAMSG1,DISP=(SHR,PASS)
//CUSTMSG2 DD DSN=&&BAMSG2,DISP=(SHR,PASS)
//CUSTSCL  DD *
   DEFINE ENVIRONMENT '&C1ELEMENT'
   TITLE 'DYNAMIC DEVELOPMENT ENVIRONMENT &C1ELEMENT'
   STAGE ONE MCF '&MCF1'
   STAGE TWO MCF '&MCF2'
   LIKE &C1ENVMNT
     NEXT ENVIRONMENT &C1ENVMNT
   .
   CLONE SYSTEM *
     FROM ENVIRONMENT &C1ENVMNT
     TO ENVIRONMENT &C1ELEMENT
     INCLUDE SUBSYSTEMS
     INCLUDE TYPES
   .
/*
//CONLIST  EXEC PGM=CONLIST,PARM=STORE,MAXRC=0,COND=EVEN
//C1LLIBO  DD   DSN=BST.ENDTEST2.ENVS.LISTLIB,
//         DISP=OLD,MONITOR=COMPONENTS,FOOTPRNT=CREATE
//C1BANNER DD   UNIT=&VIO,SPACE=(TRK,(1,1)),
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
//LIST01   DD   DSN=&&IDMSGS,DISP=(OLD,DELETE)
//LIST02   DD   DSN=&&BAMSG1,DISP=(OLD,DELETE)
//LIST03   DD   DSN=&&BAMSG2,DISP=(OLD,DELETE)
//LIST04   DD   DSN=&&ESITR,DISP=(OLD,DELETE)
```

Security Best Practices

- Access to define or modify environment is controlled at the environment level by the “PRIMARY_OPTIONS” entry in the security table (BC1TNEQU)
  - When MENUITEM is included in the PRIMARY_OPTIONS entry it resolves to a value for the primary option selected at run time, which for the ENVIRONMENT option is “ENVRMENT”
  - For example, we have this entry for PRIMARY_OPTIONS:

```
NAMEQU PRIMARY_OPTIONS,                    +
      L1=('C1'),                           +
      L2=(ENVIRONMENT),                    +
      L3=('PMENU'),                        +
      L4=(MENUITEM)
```

  - Request to define dynamic environment “WRK1” would result in this pseudo data set name check for read access: C1.WRK1.PMENU.ENVRMENT
  - If the user can read C1.WRK1.PMENU.ENVRMENT, they can define, update or delete the WRK1 Dynamic Environment and all systems, subsystems, etc.
- Consider using a prefix or range of values that can be used for defining dynamic environments and creating a profile or standard access
  - For example, granting read access to this group for data set “C1.W%%%.PMENU.ENVRMENT”

Security Best Practices

- We recommend you incorporate the Dynamic environment name into all data sets and definitions that pertain to it. Includes:
  - VSAM MCF files for the environment
  - Base, Delta, Type, Source Output and Processor output files.
- This should make it easy to identify all of the data sets that belong to a dynamic environment and to establish rules to secure these data sets and associated Endevor actions.
- For example, the base library for a type definition could be something like this, including the system, subsystem, environment, stage number and type in the data set name:
  - NDV.&C1SY..&C1SU..&C1EN..S&C1S#..&C1ELTYPE

Cleaning Up Old Environments

- You may want to clean up old environments that are no longer in use
- Leveraging the CSV utility to find a list of dynamic environments
- Then list all elements that have been updated after a given time
  - If the list is empty there have been no updates
  - Then the environment should be cleaned up
- Remember - before deleting the environment, all elements should be removed
  - And the users should also be warned!

Demonstration – Dynamic Environments and DFC

BROADCOM

Summary

- Dynamic Environments supports Agile by allowing developers to work in isolation and in parallel **regardless of how developers choose to work**
- Creating environments is a much lighter lift administratively
- Unnecessary disk usage is avoided with Deferred File Creation
- Add self-service to allow developers to create environments as needed
- Use Naming Conventions to allow security profiles to be easily managed

Questions?