Managing JCL changes through the Development Life Cycle – a different approach

Christian Le Fils and Chris Reed
ASG

November 2019
Session ML

Place your custom session QR code here. Please remove the border and text beforehand.

Managing JCL changes through the Development Life Cycle – a different approach

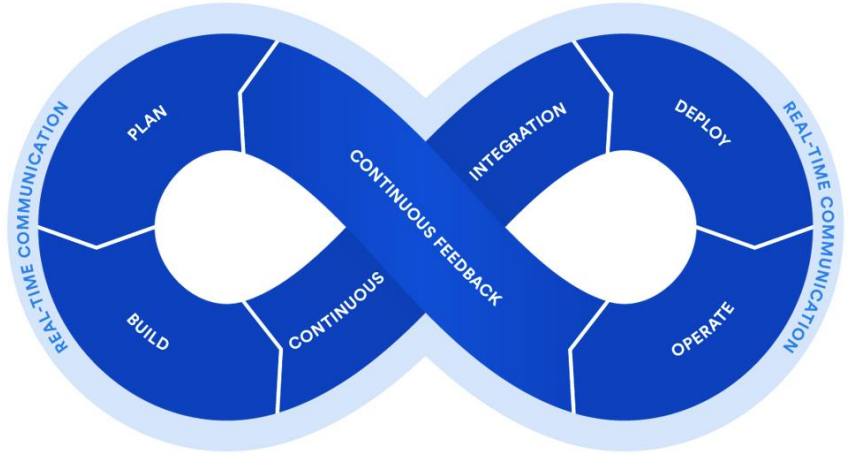
- Setting the Scene – the challenges of Managing JCL
- An Integrated and Automated Approach
- A Radical Approach – No JCL!
- Conclusions

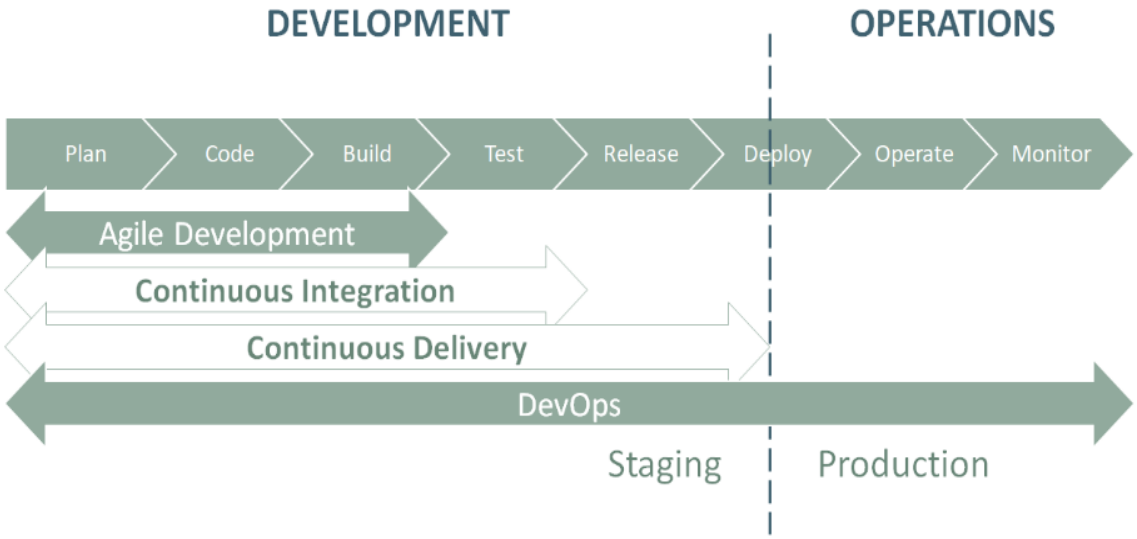
Setting the Scene –
The challenges of Managing JCL

Initiatives in a Mainframe World

DEVOPS

- DevOps
  The set of practices that emphasize the collaboration and communication of developers, testers, IT, automating the process of delivery and infrastructure changes
- DevOps Toolchain
  The products or “tools” that work together to enable the DevOps philosophy
- CI/CD Solutions
  Continuous Integration/Continuous Delivery solutions are part of the DevOps Toolchain



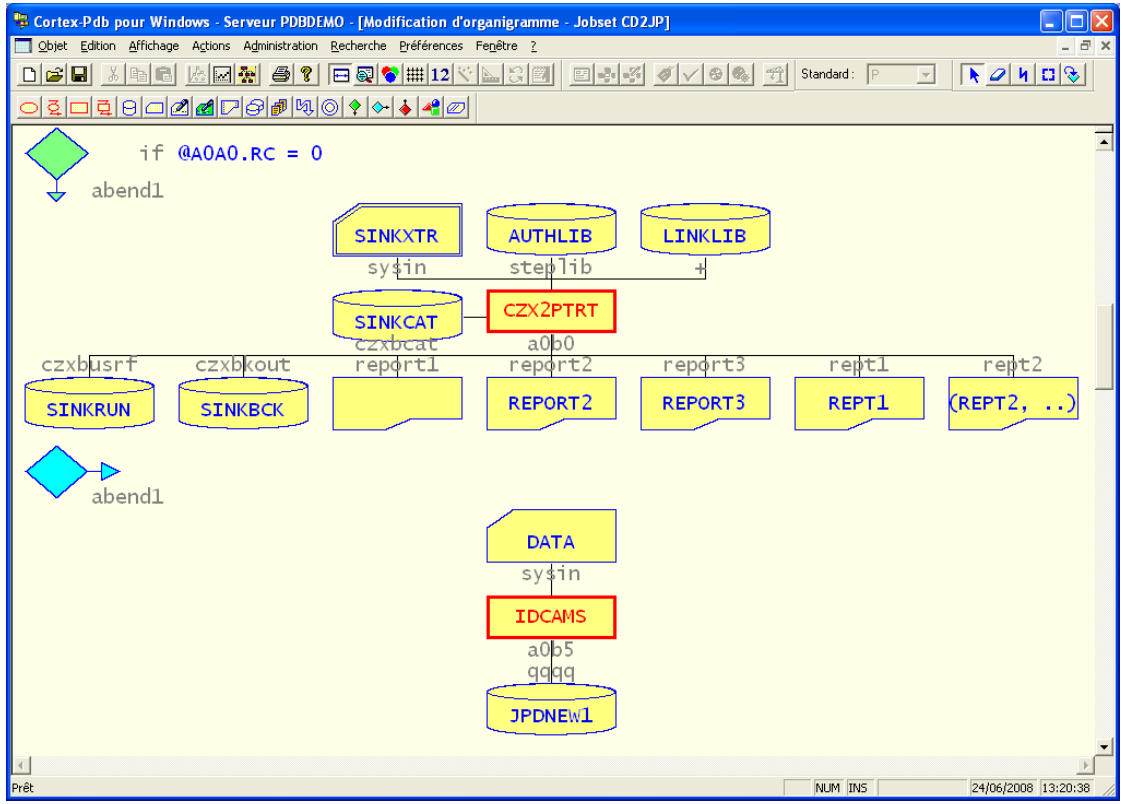


What is the state of the JCL in your Data Centre?

- How many mainframes running JCL?
- JCL running on multiple LPARs?
- JCL running in DEV, SYST, UAT etc?
- Parallel development with multiple DEV, SYST, UAT environments?
- JCL automated with one or more automation solutions?
- Developers still working on a 3270 emulators?

What is a JCL stream

- JCL is the “glue” that connects programs and files on z/OS.
- The language to run a sequence of application programs:
  - reading a bunch of sequential files
  - updating some databases
  - extracting data from those databases
  - building some new sequential files and new batch reports
- More or less something like :

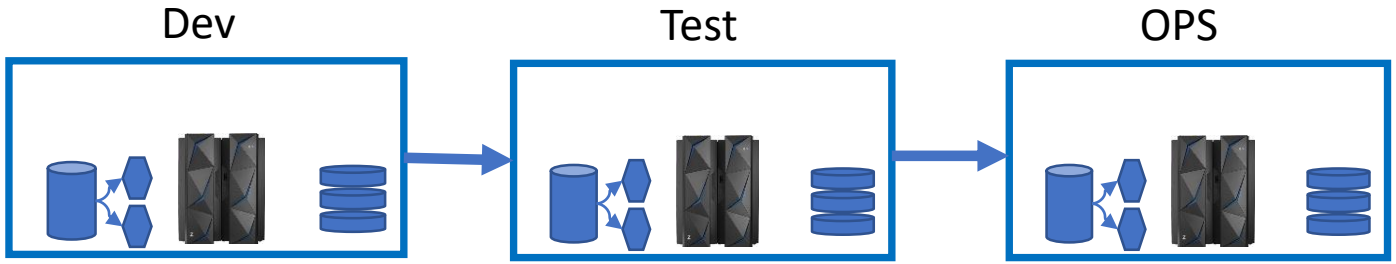


What is a JCL stream for Operations staff ?

- **The language to run a sequence of application programs, but also** :
  - Some technical steps to save the database before update, notify the scheduler, facilitate the restart in case of failure, …
  - Implementing the datasets with the constraints of the production
  - Adding some scheduling specific statements
  - Implementing GDG …
- **More or less something like :**

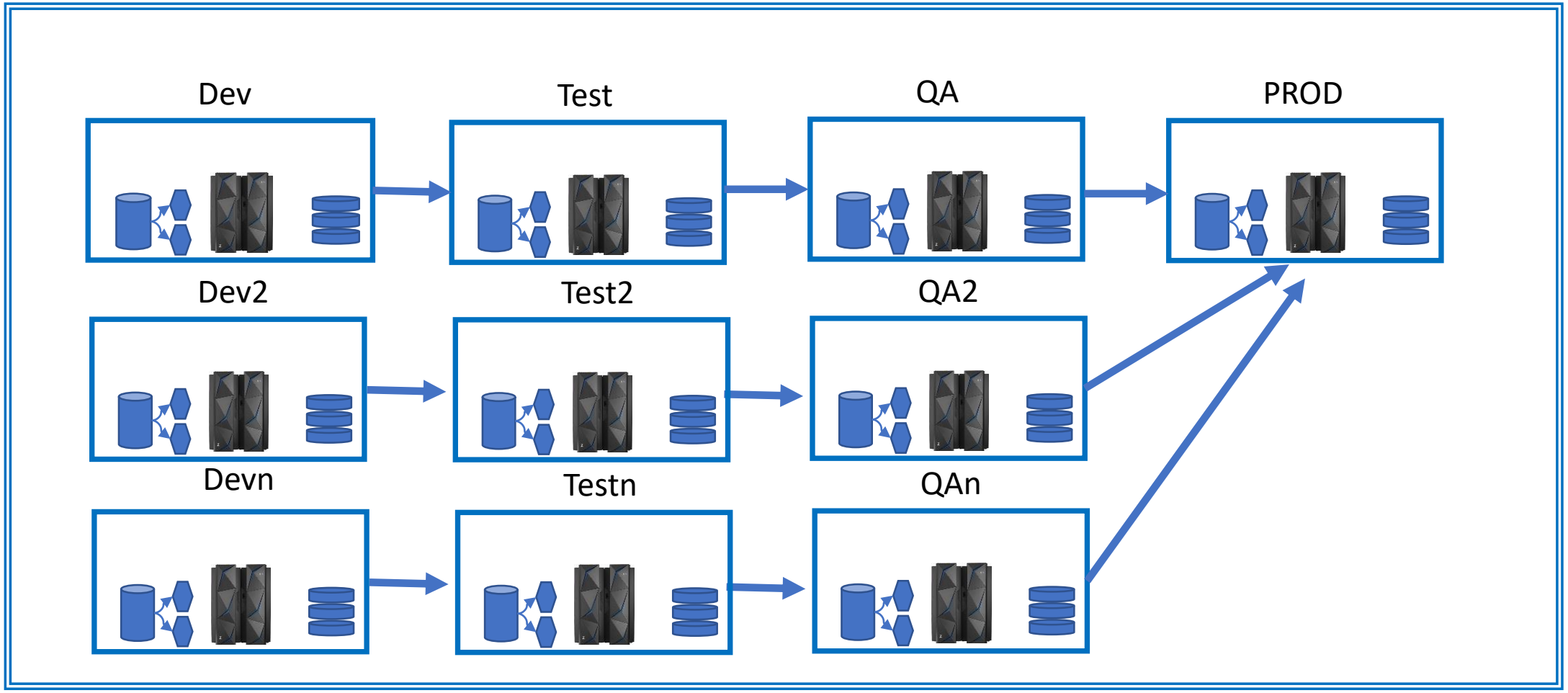
```
//UPDTA5B0 EXEC PGM=PGKSUP
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=X
//SORTF    DD   DSN=PUPDT.XP.UPDTA5A5.STRECRDS(&Gà1),DISP=SHR
//KSDSUPS  DD   DSN=PUPDT.XS.UPDTA5B0.UPDTSMMR(+1),
//              DISP=(NEW,CATLG,DELETE),DSORG=PS,
//              RECFM=FB,LRECL=50,VOL=(,RETAIN),LABEL=(RETPD=0010)
//KSDSF    DD   DSN=P.DM.KSDSFILE,DISP=SHR
//SNAPDD   DD   SYSOUT=A
//*************************************
//**        BACKUP OF KSDS           **
//*************************************
//UPDTA5B5 EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//KSDSF    DD   DSN=P.DM.KSDSFILE,DISP=SHR
//KSDSBAC  DD   DSN=PUPDT.BU.UPDTA5B5.KSDSBCKP(+1),
//              DISP=(NEW,CATLG,DELETE),DSORG=PS,
//              RECFM=FB,LRECL=80,LABEL=(RETPD=0021)
//SYSIN    DD   DSN=CORTEX.CLF.P.MTXTPRM1(UPDTA5B5),DISP=SHR
```

Is JCL the same from Dev to Ops ?



- The applicative part (what we call the functional part) is the same in both Dev and Ops, but the Technical stuff is not the same because the Production constraints are not the same.
- There are usually more strict rules in Production :
  - Dataset naming conventions, job and step naming conventions …
  - Media (Disk, VTS, …) choice are different because volume of data is usually less in Dev
  - Usage of GDG, with a appropriate number of versions (to avoid wasting disk space)
  - Structure of Jobs (start with a header step, finish with an ending step)
  - Scheduling constraints
  - And more….

Typical Mainframe Life Cycle



What does that mean for your JCL?

CHANGE

CHANGE

CHANGE

CHANGE

```
//BKUPLLD    EXEC PGM=IDCAMS,COND=(0,NE)
//SYSPRINT   DD SYSOUT=A
//SYSOUT     DD SYSOUT=A
//IN         DD DSN=IDS.VS.A0.LLD.IDSACCT,DISP=SHR
//OUT        DD DSN=IDS.QS.A0.LLD.IDSACCT.HARILALU,DISP=(NEW,CATLG),
//           DCB=(RECFM=FB,LRECL=43,BLKSIZE=0),
//           UNIT=3390,SPACE=(CYL,(10,10),RLSE)
//SYSIN      DD *
  REPRO INFILE(IN) -
        OUTFILE(OUT) -
        REPLACE
/*
//*******************************************************************
//* @D DELETE DEFINE IDS ACCOUNT DATASET (COPY FROM INTL)           *
//* @R RUN FROM SORTLLD                                            *
//*******************************************************************
//DELVSLLD EXEC PGM=IDCAMS,REGION=2048K,COND=(0,NE)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
 DELETE IDS.VS.A0.LLD.IDSACCT CL PRG
 DEFINE CLUSTER -
           (NAME(IDS.VS.A0.LLD.IDSACCT) -
           INDEXED -
           SHAREOPTIONS(2,3) -
           NOERASE -
           RECOVERY -
           NOWRITECHECK -
           NONSPANNED) -
```

What does that mean for your JCL?

CHANGE

CHANGE

CHANGE

CHANGE

```
//DELDEFNM EXEC PGM=IDCAMS,REGION=2048K,COND=(0,NE)
//*
//SYSPRINT DD   SYSOUT=A
//SYSOUT   DD   SYSOUT=A
//SYSIN    DD   *

   DELETE  (IDS.VS.A0.LLD.IDSACCT.ALTNDX)      AIX

   DEFINE AIX                                      -
         (NAME(IDS.VS.A0.LLD.IDSACCT.ALTNDX)  -
          RELATE(IDS.VS.A0.LLD.IDSACCT)       -
          NONUNIQUEKEY                        -
          UPGRADE                             -
          SPEED                               -
          KEYS(15 10)                         -
          RECORDS (3000 1000)                 -
          SHR(2 3))

   DEFINE  PATH                                   -
         (NAME(IDS.VS.A0.LLD.IDSACCT.IDSNAME)  -
          PATHENTRY(IDS.VS.A0.LLD.IDSACCT.ALTNDX)  -
          UPDATE)

   BLDINDEX INDATASET (IDS.VS.A0.LLD.IDSACCT)  -
            OUTDATASET(IDS.VS.A0.LLD.IDSACCT.ALTNDX)

   IF MAXCC = 08 THEN SET MAXCC = 0
```

What does that mean for your JCL?

CHANGE

CHANGE

CHANGE

CHANGE

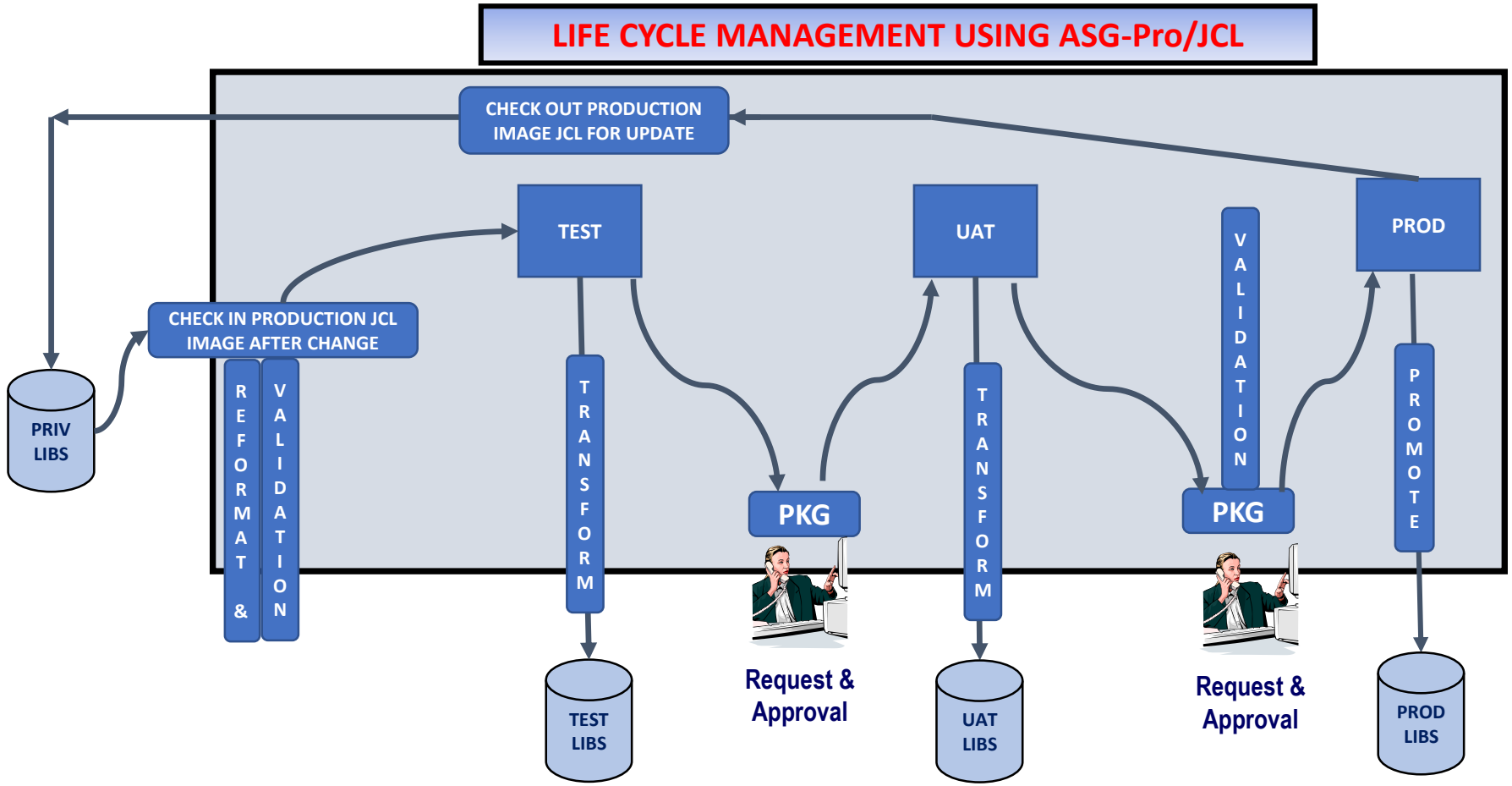
CHANGE

```
//CIBK0010 EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(0,NE)
//STEPLIB  DD DSN=DB2PRD.SDSNLOAD,DISP=SHR
//         DD DSN=MNI.BCA.P.DB2LOAD,DISP=SHR
//         DD DSN=MNI.BCA.P.LOAD,DISP=SHR
//SYSTSPRT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSOUT   DD SYSOUT=A
//SYSOUD   DD SYSOUT=A
//BULKIN   DD DSN=LKS.GS.A0.CIBK0010.BULKIN,DISP=SHR
//BANKDTE  DD DSN=IDS.QS.A0.CLDATE,DISP=SHR
//BULKCTLO DD DSN=IDS.QS.A0.CIBK0010.BULKCNTL,
//            DISP=(,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(CYL,(20,10),RLSE),
//            DCB=(RECFM=FB,LRECL=1000,BLKSIZE=0)
//CI001AGS DD DSN=IDS.GS.A0.CIBK0010.CI001AGS,
//            DISP=(,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(CYL,(20,10),RLSE),
//            DCB=(RECFM=FB,LRECL=1542,BLKSIZE=0)
//SYSTSIN  DD *
DSN SYSTEM(DSN) RETRY(0) TEST(0)
RUN PROGRAM(CIBK0010) PLAN(DPBTCHPL) -
  LIB('MNI.BCA.P.LOAD')
/*
//*-------------------------------------------------------------*
//* EOF THE CIS BATCH CUSTOMER OPEN REPORT GSAM FILES           *
```

AN INTEGRATED AND AUTOMATED APPROACH

Development Operations

Integration of JCL Management into Life Cycle Management Software



Changing Workforce

JCL Reformatting

```
000001 //REFJCL1 JOB (A,P,GL),'MESS',CLASS=P,MSGCLASS=H
000002 //*
000003 //PS010 EXEC PGM=IEBGENER
000004 //SYSPRINT DD SYSOUT=A
000005 //SYSUT1 DD DSN=TAPE9,UNIT=TAPE9,DISP=OLD
000006 //SYSUT2 DD DSN=&&TEMP,DISP=(,PASS),UNIT=SYSDA,
000007 // DCB=BLKSIZE=6160,SPACE=(TRK,(5,1))
000008 //SYSUDUMP DD SYSOUT=A
```

BEFORE REFORMATTING

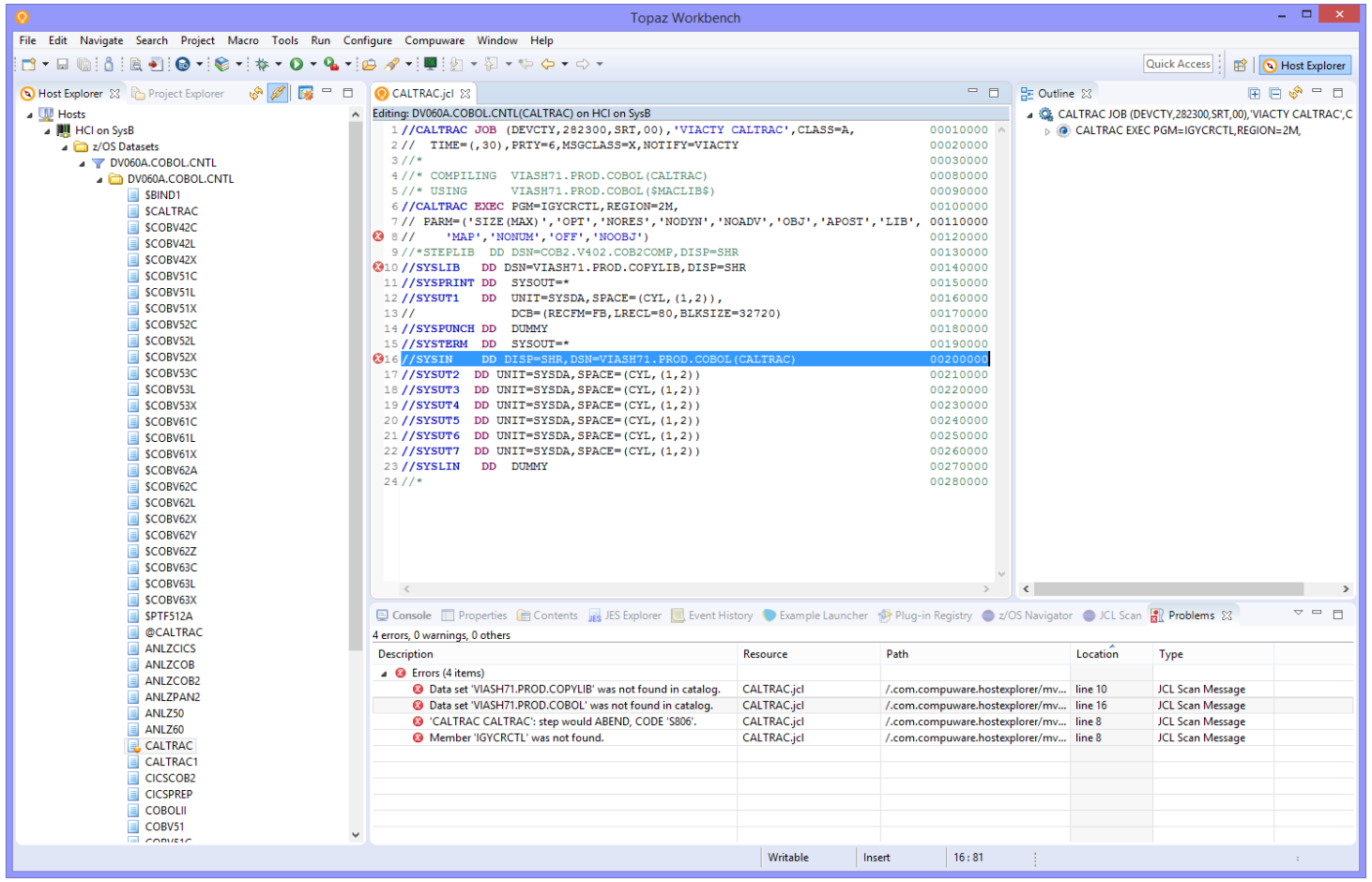
```
000001 //REFJCL1 JOB (A,P,GL),'MESS',CLASS=P,
000002 //        MSGCLASS=H
000003 //*
000004 //PS010    EXEC PGM=IEBGENER
000005 //SYSPRINT DD SYSOUT=A
000006 //SYSUT1   DD DSN=TAPE9,DISP=OLD,
000007 //            UNIT=TAPE9
000008 //SYSUT2   DD DSN=&&TEMP,DISP=(,PASS),
000009 //            UNIT=SYSDA,
000010 //            DCB=BLKSIZE=6160,
000011 //            SPACE=(TRK,(5,1))
000012 //SYSUDUMP DD SYSOUT=A
```

AFTER REFORMATTING

Changing Workforce

JCL validation

- Validation – application of standards to ensure quality and consistency & checks for errors.
- New release of Pro/JCL comes with an Eclipse front end.
- Eclipse is the foundation of popular development tools like Compuware Topaz® and IBM iDZ. Offers integration to an IDE.



JCL Transformation

Why Transformation

Production Image cannot execute in other stages. Must be modified in many place

When

Transformation takes place at the moment the JCL is written from the Life Cycle Manager tool into the JCL execution library.

How

By a fixed set of transformation rules. Transforming Production Image JCL into DEV, System Test or QA style JCL using a set of rules means that in effect the Production JCL has been validated at each stage of the life cycle. What works in Test or QA will work in Production

Transformation Rules

```
******************************** Top of Data ********************************
* TABLE 01 - CONFIG=UAT*, ALL TYPES, ALL SYSTEMS
DB2  DSN                              DSN3            BT12
DB2  DSN                              DSN2            *
DB2  DSNR                             DSN3            *
DBU  BTUSER                           BTPUSER
IMS  IMS                              IMX
IMS  IMR                              IMRX
IMS  IVP3                             IVP3X
HLQ  ATM                              ATMX
HLQ  IDS                              IDSX
HLQ  LKS                              LKSX
HLQ2 IDS.VS                           IDSX.VST
HLQ  ATMX                             ATMXX
LIB  MNI.BC.P.LOAD       LCM.ATM.UAT1.LOADLIB      MNI.BC.PCOPY.LOADLIB
****************************** Bottom of Data ******************************
```

Transformation Rules

```
****************************** Top of Data ******************************
* TABLE 01 - CONFIG=UAT*, ALL TYPES, ALL SYSTEMS
DB2   DSN                              DSN3                 BT12
DB2   DSN                              DSN2                 *
DB2   DSNR                             DSN3                 *
DBU   BTUSER                           BTPUSER
IMS   IMS                              IMX
IMS   IMR                              IMRX
IMS   IVP3                             IVP3X
HLQ   ATM                              ATMX
HLQ   IDS                              IDSX
HLQ2  LKS.GS                           LKSX.GST
HLQ   ATMX                             ATMXX
LIB   MNI.BC.P.LOAD       LCM.ATM.UAT1.LOADLIB      MNI.BC.PCOPY.LOADLIB
****************************** Bottom of Data ***************************
```

Transformation Rules

```
******************************** Top of Data ********************************
* TABLE 01 - CONFIG=UAT*, ALL TYPES, ALL SYSTEMS
DB2   DSN                                DSN3              BT12
DB2   DSN                                DSN2              *
DB2   DSNR                               DSN3              *
DBU   BTUSER                             BTPUSER
IMS   IMS                                IMX
IMS   IMR                                IMRX
IMS   IVP3                               IVP3X
HLQ   ATM                                ATMX
HLQ   IDS                                IDSX
HLQ2  LKS.GS                             LKSX.GST
HLQ   ATMX                               ATMXX
LIB   MNI.BC.P.LOAD       LCM.ATM.UAT1.LOADLIB       MNI.BC.PCOPY.LOADLIB
****************************** Bottom of Data ******************************
```

Transformation Rules

```
******************************** Top of Data ********************************
* TABLE 01 - CONFIG=UAT*, ALL TYPES, ALL SYSTEMS
DB2   DSN                              DSN3              BT12
DB2   DSN                              DSN2              *
DB2   DSNR                             DSN3              *
DBU   BTUSER                           BTPUSER
IMS   IMS                              IMX
IMS   IMR                              IMRX
IMS   IVP3                             IVP3X
HLQ   ATM                              ATMX
HLQ   IDS                              IDSX
HLQ2  LKS.GS                           LKSX.GST
HLQ   ATMX                             ATMXX
LIB   MNI.BC.P.LOAD       LCM.ATM.UAT1.LOADLIB       MNI.BC.PCOPY.LOADLIB
****************************** Bottom of Data ******************************
```

Transformation Rules

```
******************************** Top of Data ********************************
* TABLE 01 - CONFIG=UAT*, ALL TYPES, ALL SYSTEMS
DB2  DSN                                DSN3              BT12
DB2  DSN                                DSN2              *
DB2  DSNR                               DSN3              *
DBU  BTUSER                             BTPUSER
IMS  IMS                                IMX
IMS  IMR                                IMRX
IMS  IVP3                               IVP3X
HLQ  ATM                                ATMX
HLQ  IDS                                IDSX
HLQ2 LKS.GS                             LKSX.GST
HLQ  ATMX                               ATMXX
LIB  MNI.BC.P.LOAD       LCM.ATM.UAT1.LOADLIB       MNI.BC.PCOPY.LOADLIB
******************************** Bottom of Data ********************************
```

Transformation Rules

```
******************************** Top of Data ********************************
* TABLE 01 - CONFIG=UAT*, ALL TYPES, ALL SYSTEMS
DB2   DSN                                DSN3              BT12
DB2   DSN                                DSN2              *
DB2   DSNR                               DSN3              *
DBU   BTUSER                             BTPUSER
IMS   IMS                                IMX
IMS   IMR                                IMRX
IMS   IVP3                               IVP3X
HLQ   ATM                                ATMX
HLQ   IDS                                IDSX
HLQ2  LKS.GS                             LKSX.GST
HLQ   ATMX                               ATMXX
LIB   MNI.BC.P.LOAD       LCM.ATM.UAT1.LOADLIB       MNI.BC.PCOPY.LOADLIB
****************************** Bottom of Data ******************************
```

Transformation Rules

```
******************************** Top of Data ********************************
* TABLE 01 - CONFIG=UAT*, ALL TYPES, ALL SYSTEMS
DB2   DSN                              DSN3              BT12
DB2   DSN                              DSN2              *
DB2   DSNR                             DSN3              *
DBU   BTUSER                           BTPUSER
IMS   IMS                              IMX
IMS   IMR                              IMRX
IMS   IVP3                             IVP3X
HLQ   ATM                              ATMX
HLQ   IDS                              IDSX
HLQ2  LKS.GS                           LKSX.GST
HLQ   ATMX                             ATMXX
LIB   MNI.BC.P.LOAD       LCM.ATM.UAT1.LOADLIB       MNI.BC.PCOPY.LOADLIB
****************************** Bottom of Data ********************************
```

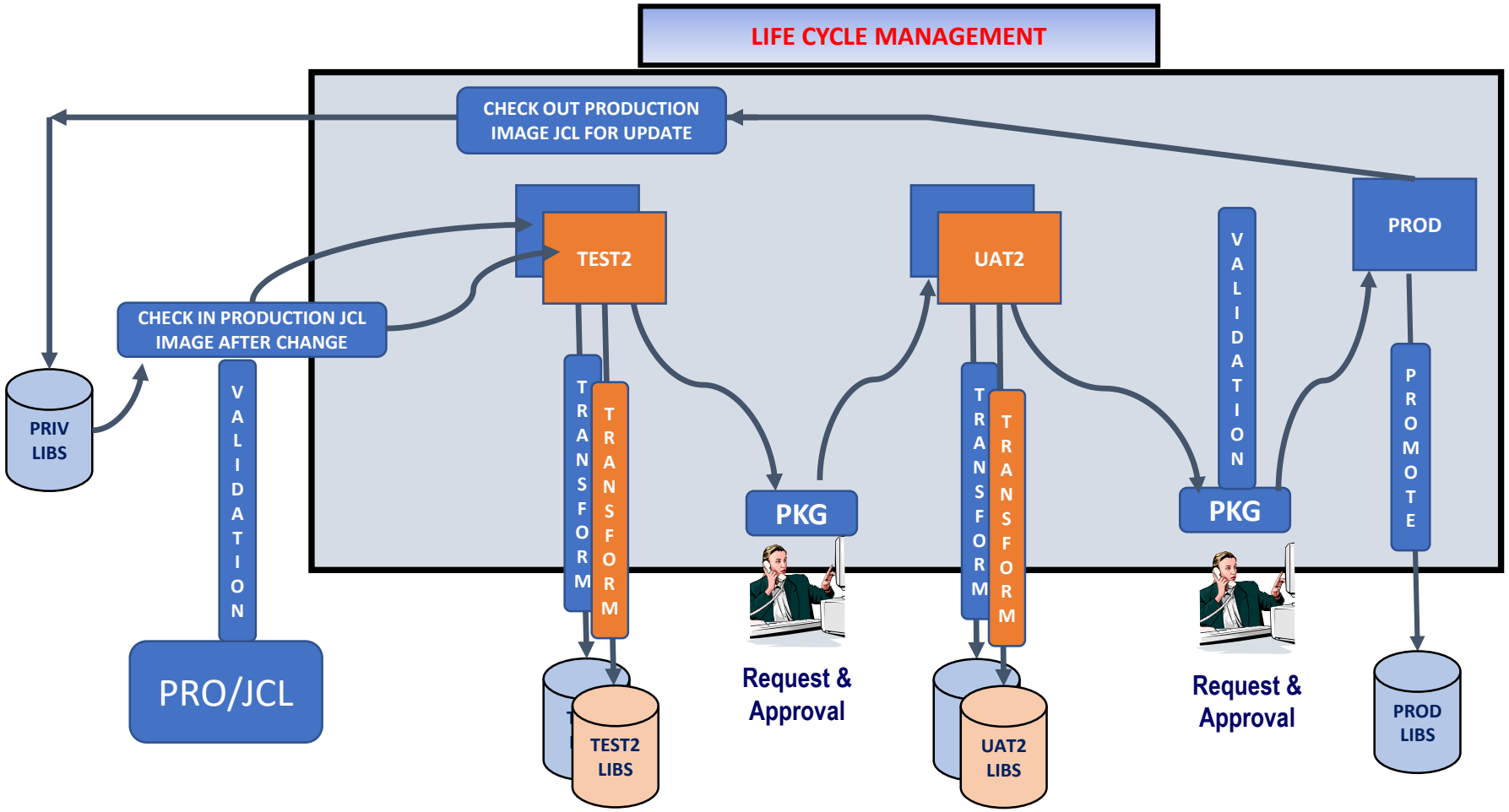
Before and after example of transformed JCL

```
//CIBK0010 EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(0,NE)
//STEPLIB  DD DSN=DB2PRD.SDSNLOAD,DISP=SHR
//         DD DSN=MNI.BCA.P.DB2LOAD,DISP=SHR
//           DD DSN=MNI.BC.P.LOAD,DISP=SHR
//SYSTSPRT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSOUT   DD SYSOUT=A
//BULKIN   DD DSN=LKS.GS.A0.CIBK0010.BULKIN,DISP=SHR
//BANKDTE  DD DSN=IDS.QS.A0.GLDATE,DISP=SHR
//BULKCTLO DD DSN=IDS.QS.A0.CIBK0010.BULKCNTL,
//            DISP=(,CATLG,DELETE),UNIT=SYSDA,
//            SPACE=(CYL,(20,10),RLSE),
//            DCB=(RECFM=FB,LRECL=1000,BLKSIZE=0)
//CI001AGS DD DSN=IDS.GS.A0.CIBK0010.CI001AGS,
//            DISP=(,CATLG,DELETE),UNIT=SYSDA,
//            SPACE=(CYL,(20,10),RLSE),
//            DCB=(RECFM=FB,LRECL=1542,BLKSIZE=0)
//SYSTSIN  DD *
DSN SYSTEM(DSN) RETRY(0) TEST(0)
RUN PROGRAM(CIBK0010) PLAN(DPBTCHPL) -
  LIB('MNI.BC.P.LOAD')
/*
//*-------------------------------------------------*
//* EOF THE CIS BATCH CUSTOMER OPEN REPORT GSAM FILES
```

```
//CIBK0010 EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(0,NE)
//STEPLIB  DD DSN=DB2PRD.SDSNLOAD,DISP=SHR
//         DD DSN=MNI.BCA.P.DB2LOAD,DISP=SHR
//           DD DSN=LCM.ATM.UAT1.LOADLIB,DISP=SHR
//           DD DSN=MNI.BC.PCOPY.LOADLIB,DISP=SHR
//SYSTSPRT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSOUT   DD SYSOUT=A
//BULKIN   DD DSN=LKSX.GST.A0.CIBK0010.BULKIN,DISP=SHR
//BANKDTE  DD DSN=IDSX.QS.A0.GLDATE,DISP=SHR
//BULKCTLO DD DSN=IDSX.QS.A0.CIBK0010.BULKCNTL,
//            DISP=(,CATLG,DELETE),UNIT=SYSDA,
//            SPACE=(CYL,(20,10),RLSE),
//            DCB=(RECFM=FB,LRECL=1000,BLKSIZE=0)
//CI001AGS DD DSN=IDSX.GS.A0.CIBK0010.CI001AGS,
//            DISP=(,CATLG,DELETE),UNIT=SYSDA,
//            SPACE=(CYL,(20,10),RLSE),
//            DCB=(RECFM=FB,LRECL=1542,BLKSIZE=0)
//SYSTSIN  DD *
DSN SYSTEM(DSN2) RETRY(0) TEST(0)
RUN PROGRAM(CIBK0010) PLAN(DPBTCHPL) -
  LIB('LCM.ATM.UAT1.LOADLIB','MNI.BC.PCOPY.LOADLIB')
/*
//*-------------------------------------------------------*
//* EOF THE CIS BATCH CUSTOMER OPEN REPORT GSAM FILES
```

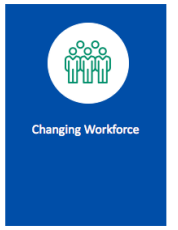
Agile Environment

Integration of JCL Management Supporting Agile development

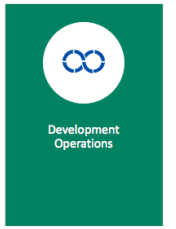


Bring JCL Changes in from the cold!

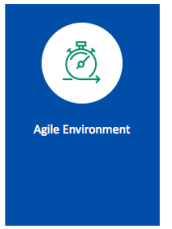
- Easier to use. Accessible from Eclipse based development workbenches in common with many other development tools.



- Automation of changes for each stage of the life cycle
  - You are sure the constraints of each stage (Dev, QA, Preprod, Prod) are **strictly** followed
  - Everybody **works the same way**, avoiding loss of time when an other person has to check or maintain a JCL
  - You gain **productivity**, because adding technical steps or transforming a dataset name is not productive and prone of errors



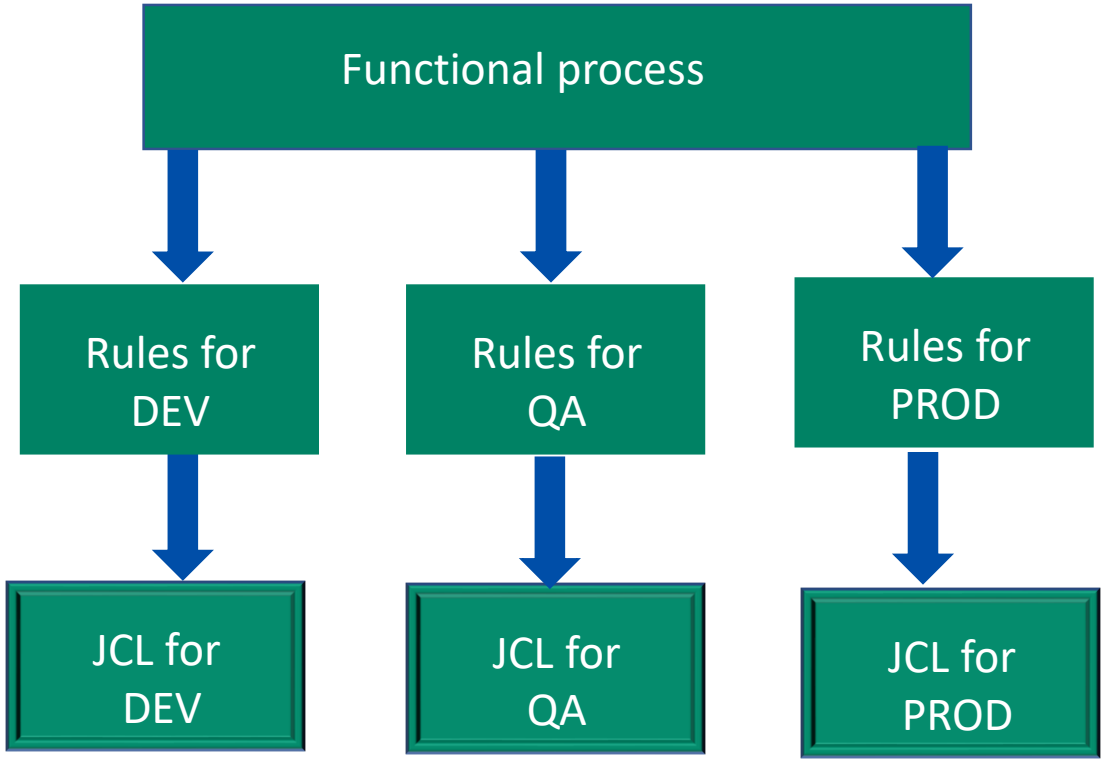
- Easier to work with parallel development environments, frees the developer from needing to know all the technical constraints of each development environment.
- Supports development on IBM Z Development and Test Environment

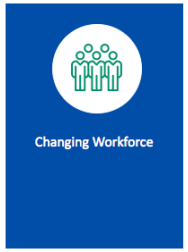


A Radical Approach – No JCL!

A Radical Approach – No JCL!

- The ASG-Cortex PDB approach is that JCL is the transposition of a functional process into a technical instance in a designated technical environment
- The idea is that the path from functional description of the process to the technical JCL stream can be automated by applying some customized rules
- So, by applying different rule sets, you can also get different JCL streams: one for DEV, one for QA, one for PROD





A Functional Language (PCL) - to describe the process

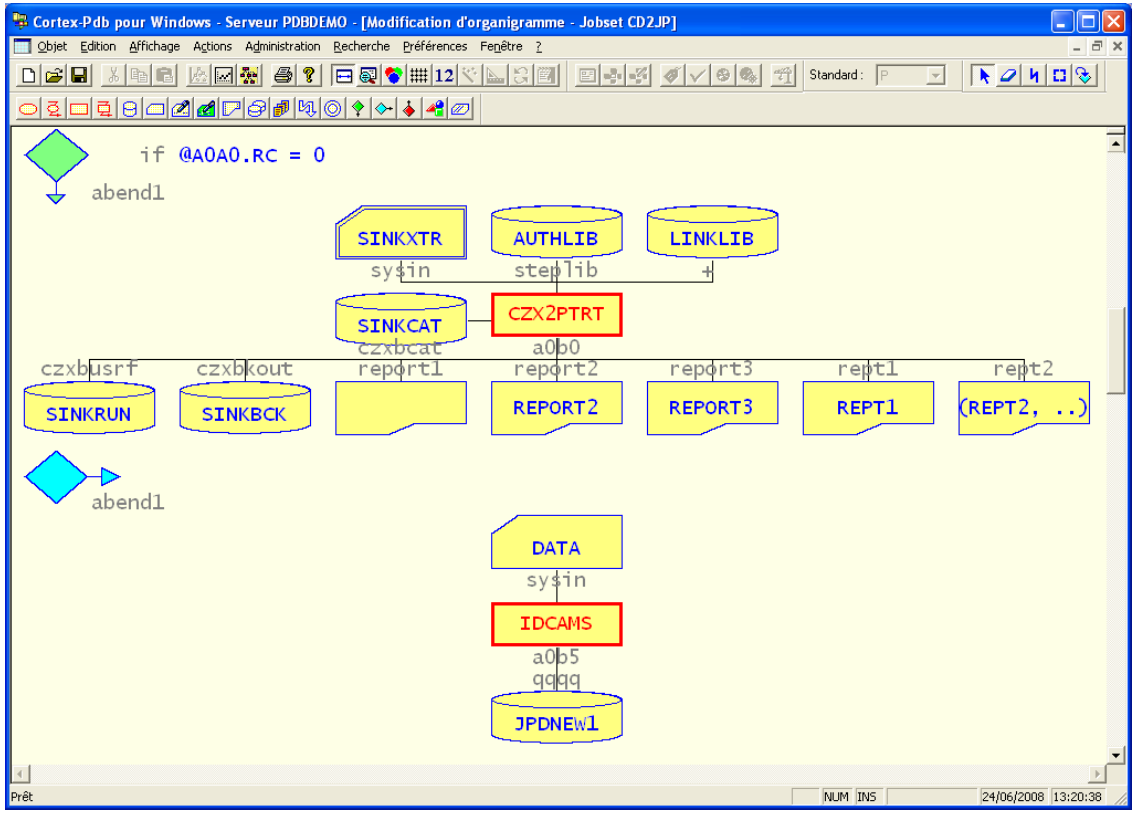
- To allow functional description, ASG-Cortex PDB embeds a functional and concise language called PCL ('Production Control Language')
- In the example on right, you can see that the access mode to a file is more precise than the DISP parameter in the JCL
- From that language easy to learn, ASG-Cortex PDB is able to generate the different 'flavours' of JCL

```
 STEP  PGM=LKRB0007,LANG=CBL
*
LKL6600  FILE  MODE=I,NAME=LKL6600
LKR0801  FILE  MODE=I,NAME=LKL0801
LKM8301  FILE  MODE=O,NAME=LKL83AA
LKM8302  FILE  MODE=O,NAME=LKL83AB
LOM7131  FILE  MODE=O,NAME=LKL7131
LKC1001  FILE  MODE=U,NAME=LKL1046
LOP0301  FILE  MODE=O,NAME=LKL03PW
LKR0008L REPORT NAME=LBI0015R
COMREG   DATA  *
//*** MESSAGE      STE-COD 0128  --> REMA.
      DATAEND
LOP0101  DATA  *
STE-COD 0128
      DATAEND
```

Changing Workforce

Not Excited to Learn a New Language?

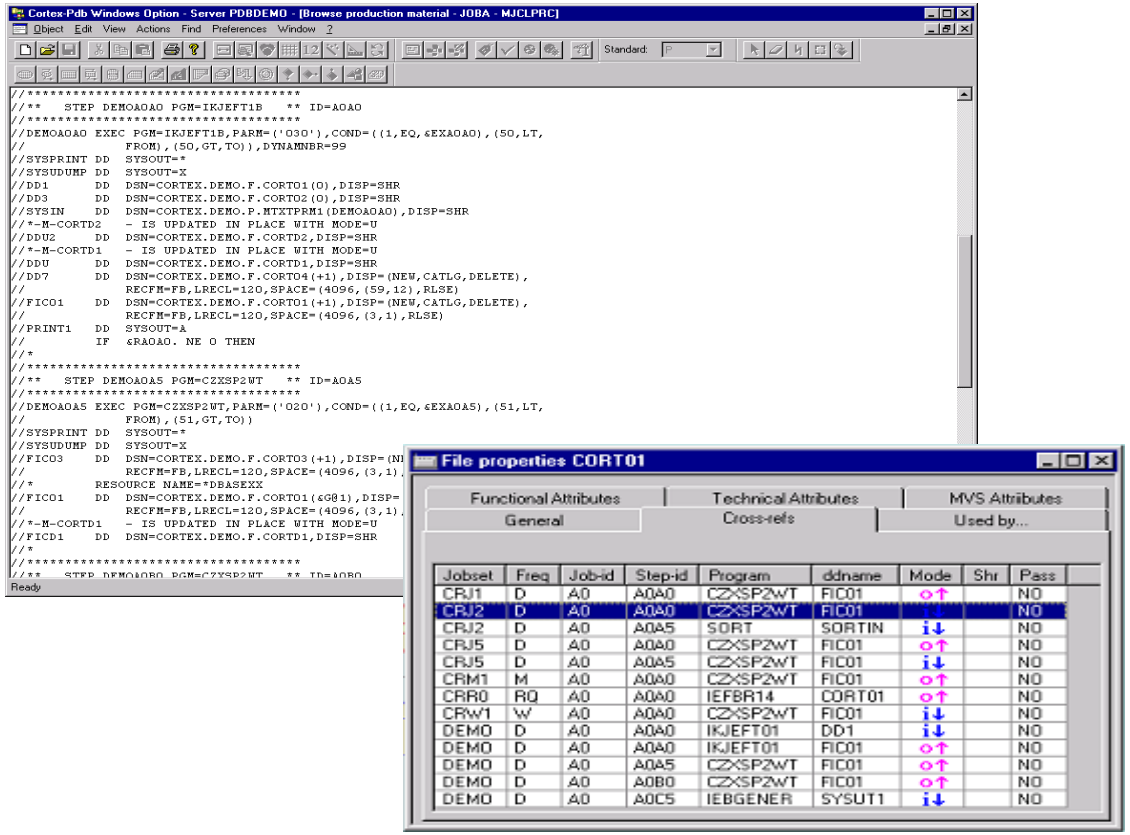
- New people in the team have not a lot of knowledge in mainframe and JCL
- Hope they will like this kind of tool to describe the functional process.
- From a Windows station connected to the mainframe server, using Drag and Drop facilities, they can draw the process as they would do on a paper.
- When they save the flowchart, PCL stream is created in the mainframe.



Get a JCL Stream Through a Single Click !

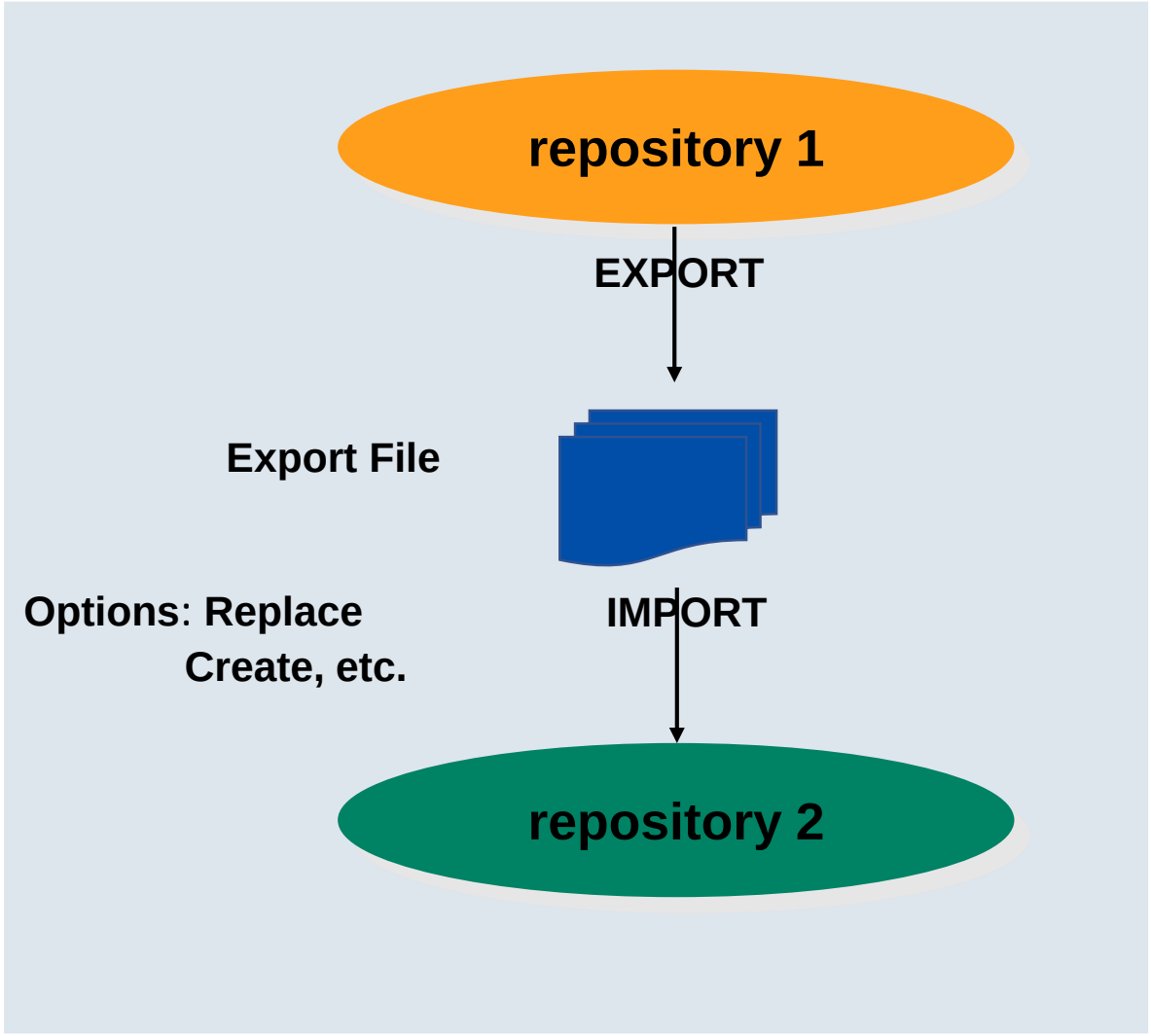
- From the same tool; a user can also:
  - Push a button to start the JCL generation (on mainframe) and get the result in the Windows tool
  - Launch requests like “where is my file used ?”
  - Print the flowchart

PROD

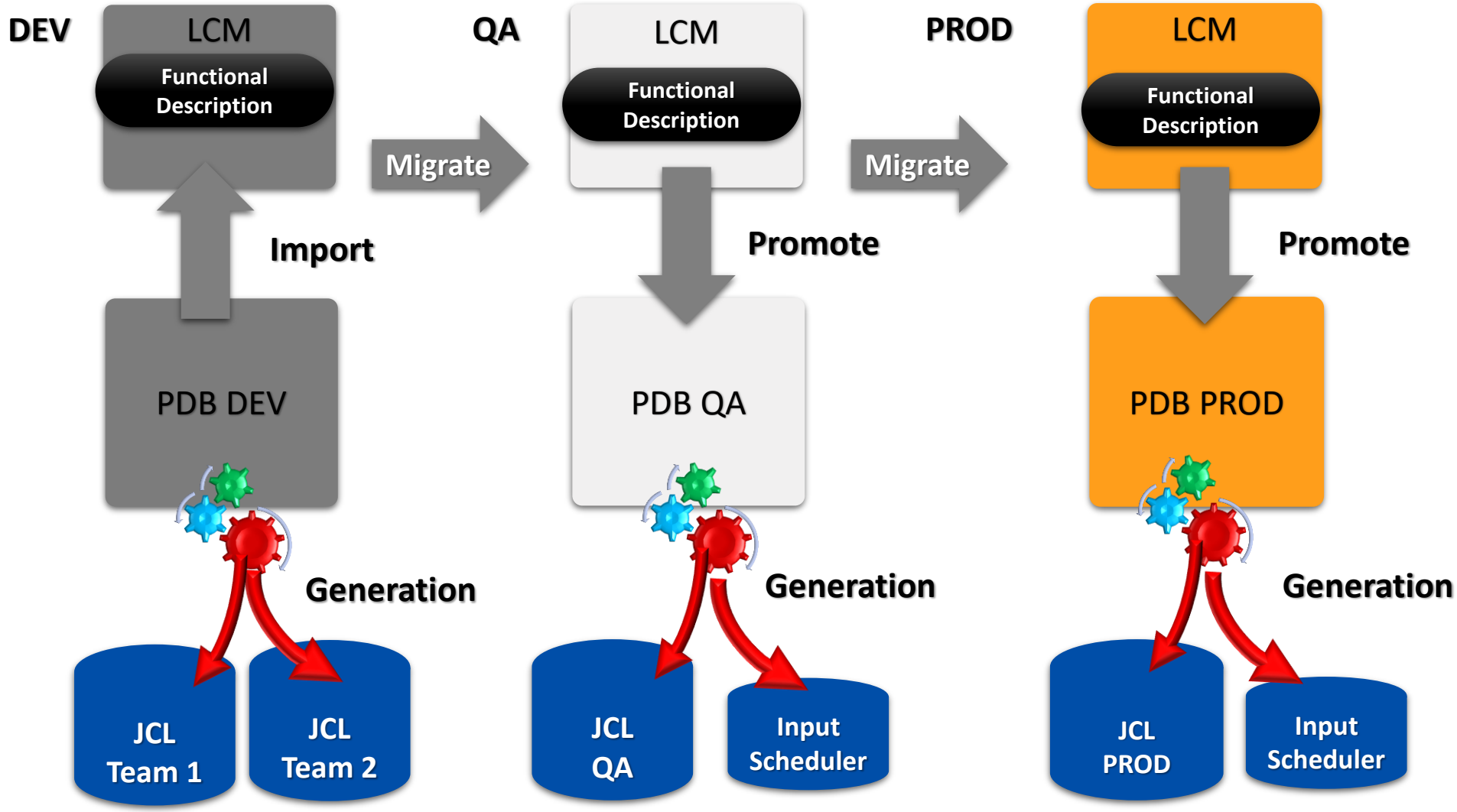


Architecture Flexibility

- ASG-Cortex PDB allows different architectures:
  - Several set of rules in one unique PDB repository (so called ‘standards’)
  - Several PDB repositories exchanging functional descriptions (PCL packages)
  - Mix of multiple repositories and multiple set of rules
  - One repository can support 1 to 26 set of rules
- Tools Import and Export allows to exchange PCL from one PDB to another PDB
- Exchanges can be controlled through a Life Cycle Manager (LCM, ENDEVOR, CHANGEMAN, …)

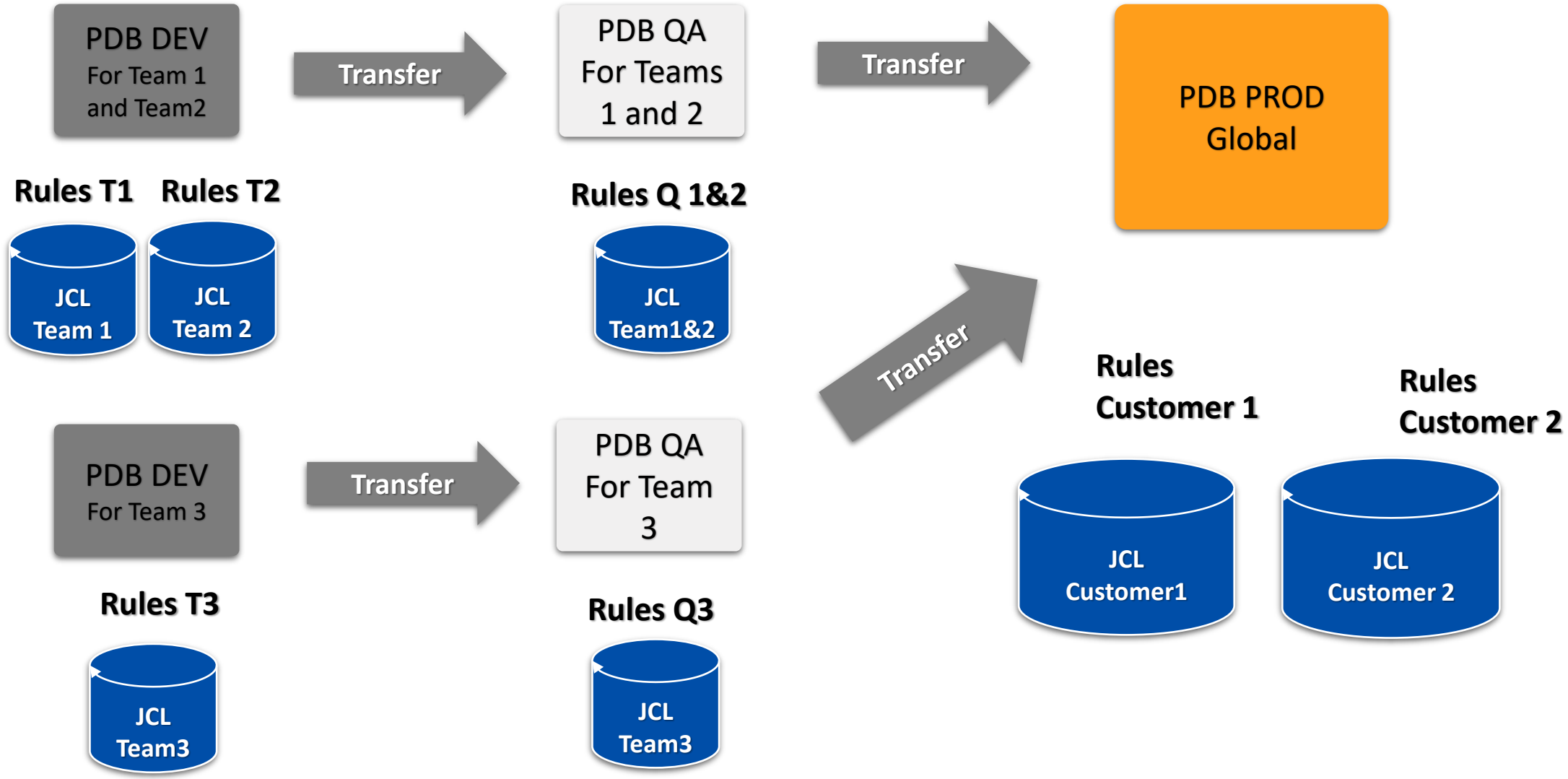


Organize the DEVOPS lifecycle



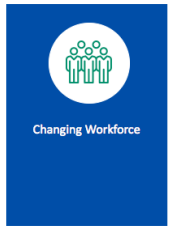
Agile Environment

Facilitate Agile working



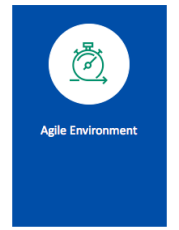
Why generation of JCL is a modern approach

- Your staff do not require JCL coding skills



- You get an easier and faster way to go from Dev to Ops
- You are sure the constraints of each stage (Dev, Qualification, Preprod, Prod) are **strictly** followed
- Everybody **works the same way**, avoiding loss of time when an other person has to check or maintain a JCL
- You gain **productivity**, because adding technical steps or transforming a dataset name is not productive and prone of errors

- If you work for different customers with same application, you can generate the JCL ad hoc for each one of them



And in Conclusion…..

Initiatives in a Mainframe World

GUIDE SHARE EUROPE UK REGION

THANK YOU!

Discussion & Questions?

Please submit your session feedback!

- Do it online at http://conferences.gse.org.uk/2019/feedback/ML
- This session is ML

Place your custom session QR code here. Please remove the border and text beforehand.