Smart Building with IBM DBB

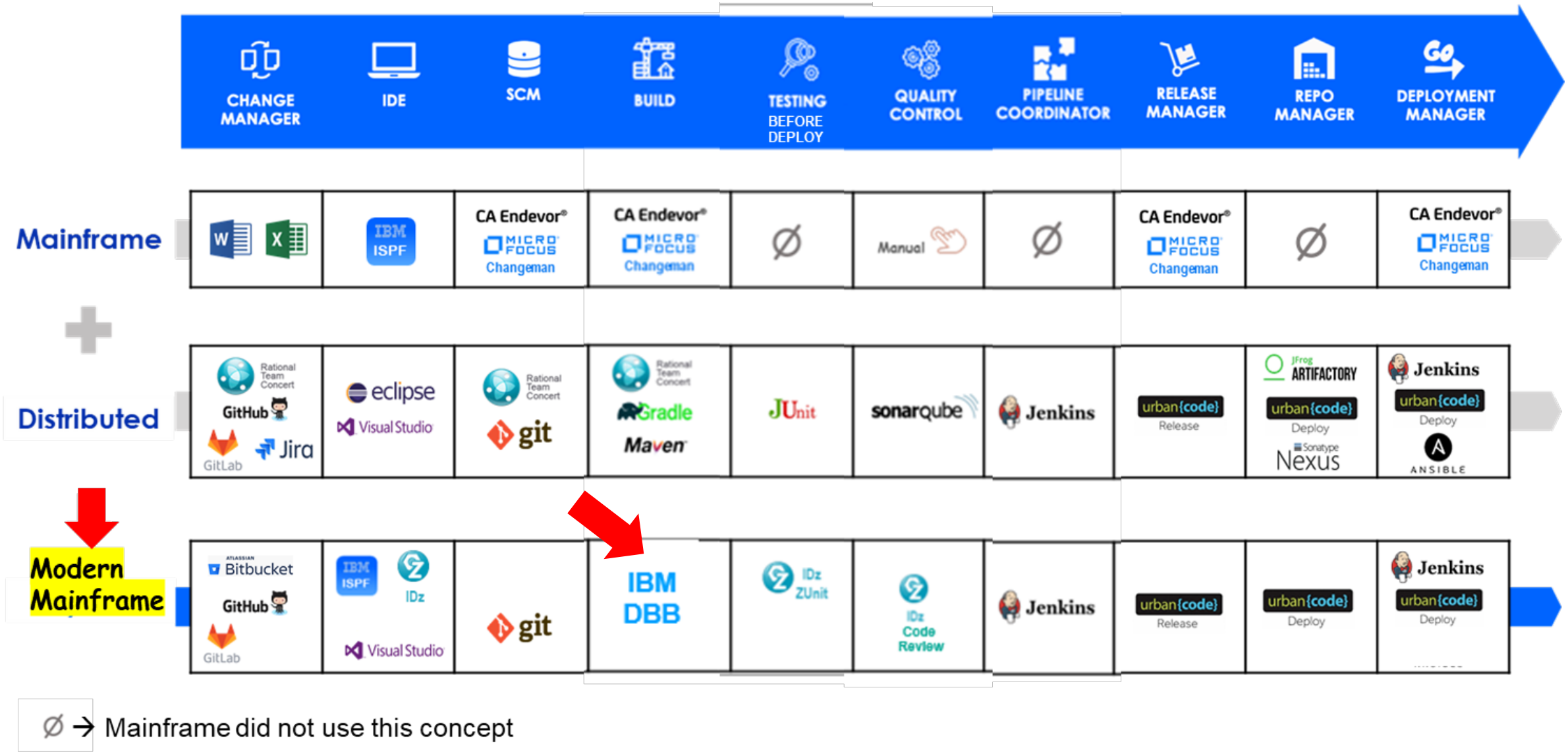
How to improve Application Development on Z/OS

Reginaldo Barosa
IBM USA - Executive IT Specialist

On this session we will show how IBM DBB can help on the traditional z/OS development using COBOL or PL/I as language.
IBM DBB allows to interact with z/OS via APIs instead of the traditional JCL and also provides a framework that helps application development and z/OS build (compile/link/bind) using any SCM of your choice.
The DBB framework provides an Application Server (usually on Linux) that stores all the dependencies of the source code (like copybooks). Also DBB can stores all the build results.
DBB allow z/OS to implement what is usually called "Smart Building".
We also will show how DBB integrates with Open Source tools like Jenkins and Git in the z/OS environment.

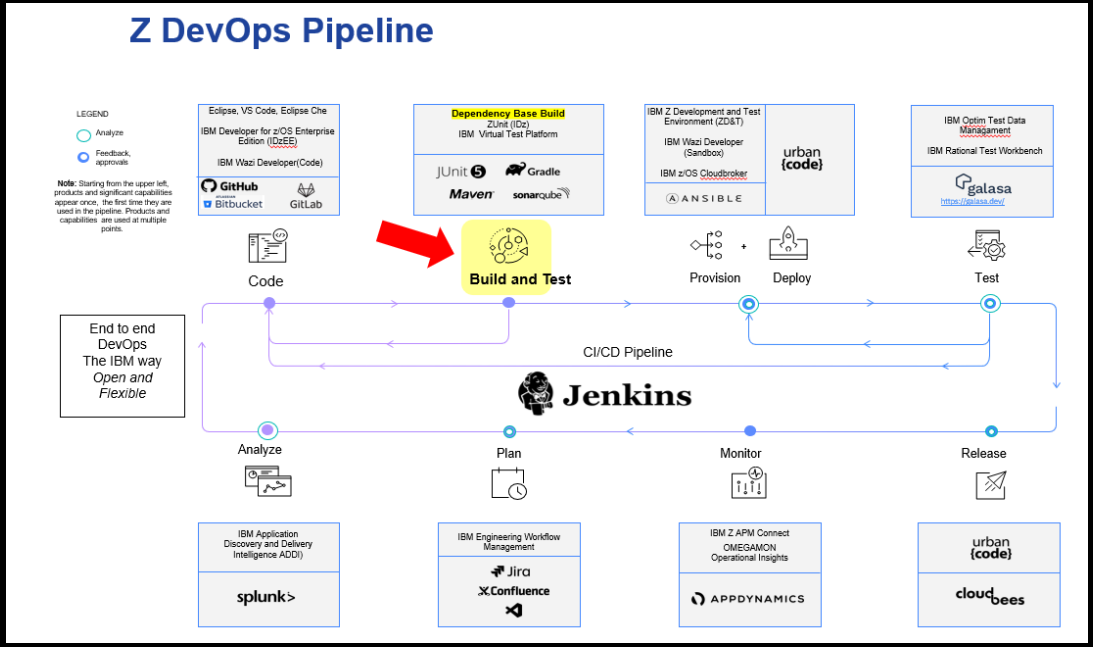
Nesta sessão iremos mostrar como IBM DBB no ambiente z/OS auxilia na programação tradicional com COBOL ou PL/I.
IBM DBB permite executar funções no z/OS usando APIs ao invés de JCL e fornece um framework para entender e executar builds no ambiente z/OS com qualquer SCM de sua escolha.
Junto com este framework um Servidor Web armazena as dependências e os resultados dos vários builds, permitindo o que podemos chamar de "Smart Building".
Nesta sessão iremos mostrar como funciona o DBB e como se integra num ambiente com ferramentas de Open Source como **Jenkins** e **Git** no ambiente z/OS.

Imagine a unified Pipeline on Z/OS



What is

IBM’s Dependency Based Build (DBB)?



DBB is an automated Build Tool for mainframe applications – no *JCL, ISPF, SDSF* needed

It's a stand-alone product that integrates with any modern IDE or SCM like Git

Enables automation through modern scripting languages like Groovy & Python under any pipeline Orchestrator like Jenkins

Example Compile JCL vs Groovy + DBB APIs

```
//COBOL  EXEC PGM=IGYCRCTL,REGION=0M,
//       PARM='LIB'
//*
//STEPLIB  DD DISP=SHR,DSN=IGY.SIGYCOMP
//*
//SYSIN    DD DISP=SHR,DSN=USER1.BUILD.COBOL(HELLO)
//SYSLIN   DD DISP=SHR,DSN=USER1.BUILD.OBJ(HELLO)
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DISP=(NEW,PASS),SPACE=(TRK,(5,5)),UNIT=VIO,
//            BLKSIZE=80,LRECL=80,RECFM=FB
//SYSUT2   DD DISP=(NEW,PASS),SPACE=(TRK,(5,5)),UNIT=VIO,
//            BLKSIZE=80,LRECL=80,RECFM=FB
//SYSUT3   DD DISP=(NEW,PASS),SPACE=(TRK,(5,5)),UNIT=VIO,
//            BLKSIZE=80,LRECL=80,RECFM=FB
//SYSUT4   DD DISP=(NEW,PASS),SPACE=(TRK,(5,5)),UNIT=VIO,
//            BLKSIZE=80,LRECL=80,RECFM=FB
//SYSUT5   DD DISP=(NEW,PASS),SPACE=(TRK,(5,5)),UNIT=VIO,
//            BLKSIZE=80,LRECL=80,RECFM=FB
//SYSUT6   DD DISP=(NEW,PASS),SPACE=(TRK,(5,5)),UNIT=VIO,
//            BLKSIZE=80,LRECL=80,RECFM=FB
//SYSUT7   DD DISP=(NEW,PASS),SPACE=(TRK,(5,5)),UNIT=VIO,
//            BLKSIZE=80,LRECL=80,RECFM=FB
```

```
import com.ibm.dbb.build.*

println("Copying source from zFS to PDS . . .")
def copy = new CopyToPDS()
              .file(new File("/u/usr1/build/helloworld.cbl"))
              .dataset("USR1.BUILD.COBOL").member("HELLO")
copy.execute()

println("Compiling . . .")
def compile = new MVSExec().pgm("IGYCRCTL").parm("LIB")
compile.dd(new DDStatement().name("TASKLIB")
              .dsn("IGY.SIGYCOMP").options("shr"))
compile.dd(new DDStatement().name("SYSIN")
              .dsn("USR1.BUILD.COBOL(HELLO)").options("shr"))
compile.dd(new DDStatement().name("SYSLIN")
              .dsn("USR1.BUILD.OBJ(HELLO)").options("shr"))
compile.dd(new DDStatement().name("SYSPRINT")
              .options("tracks space(5,5) unit(vio) new"))
compile.dd(new DDStatement().name("SYSUT1")
              .options("tracks space(5,5) unit(vio) new"))
compile.dd(new DDStatement().name("SYSUT2")
              .options("tracks space(5,5) unit(vio) new"))
compile.dd(new DDStatement().name("SYSUT3")
              .options("tracks space(5,5) unit(vio) new"))
compile.dd(new DDStatement().name("SYSUT4")
              .options("tracks space(5,5) unit(vio) new"))
compile.dd(new DDStatement().name("SYSUT5")
              .options("tracks space(5,5) unit(vio) new"))
compile.dd(new DDStatement().name("SYSUT6")
              .options("tracks space(5,5) unit(vio) new"))
compile.dd(new DDStatement().name("SYSUT7")
              .options("tracks space(5,5) unit(vio) new"))
compile.copy(new CopyToHFS().ddName("SYSPRINT")
              .file(new File("/u/usr1/build/helloworld.log")))
def rc = compile.execute()

if (rc > 4)
    println("Compile failed!  RC=$rc")
else
    println("Compile successful!  RC=$rc")
```


Demonstration

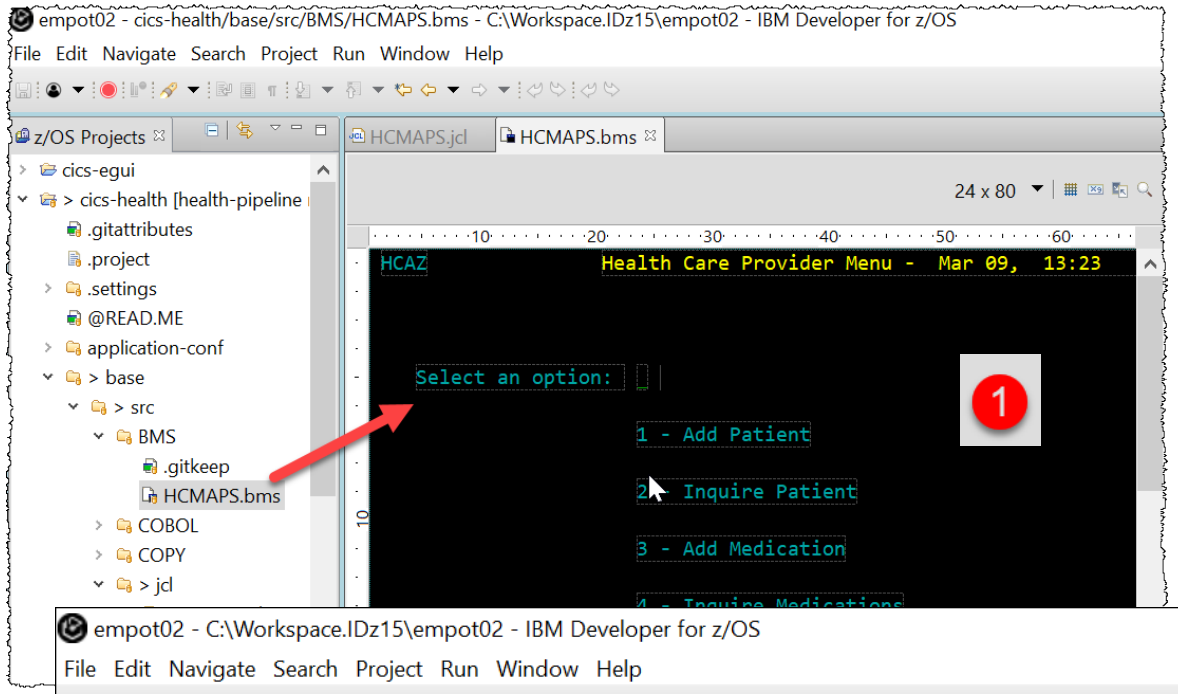
CICS/DB2 COBOL Program

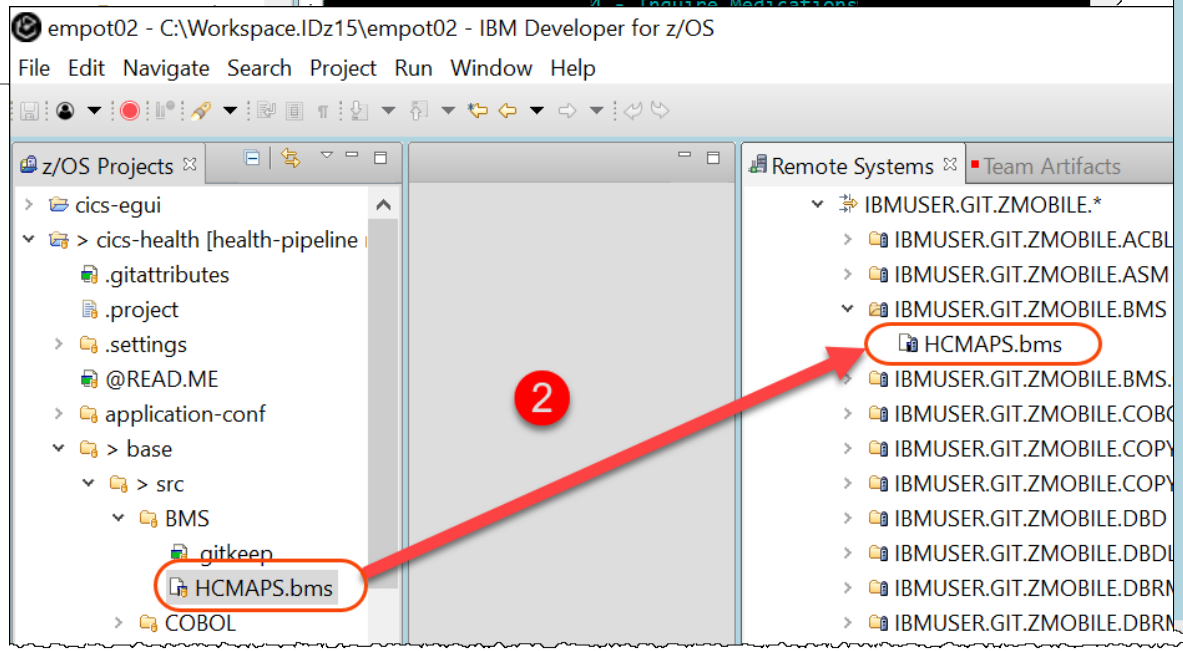
DBB versus Traditional

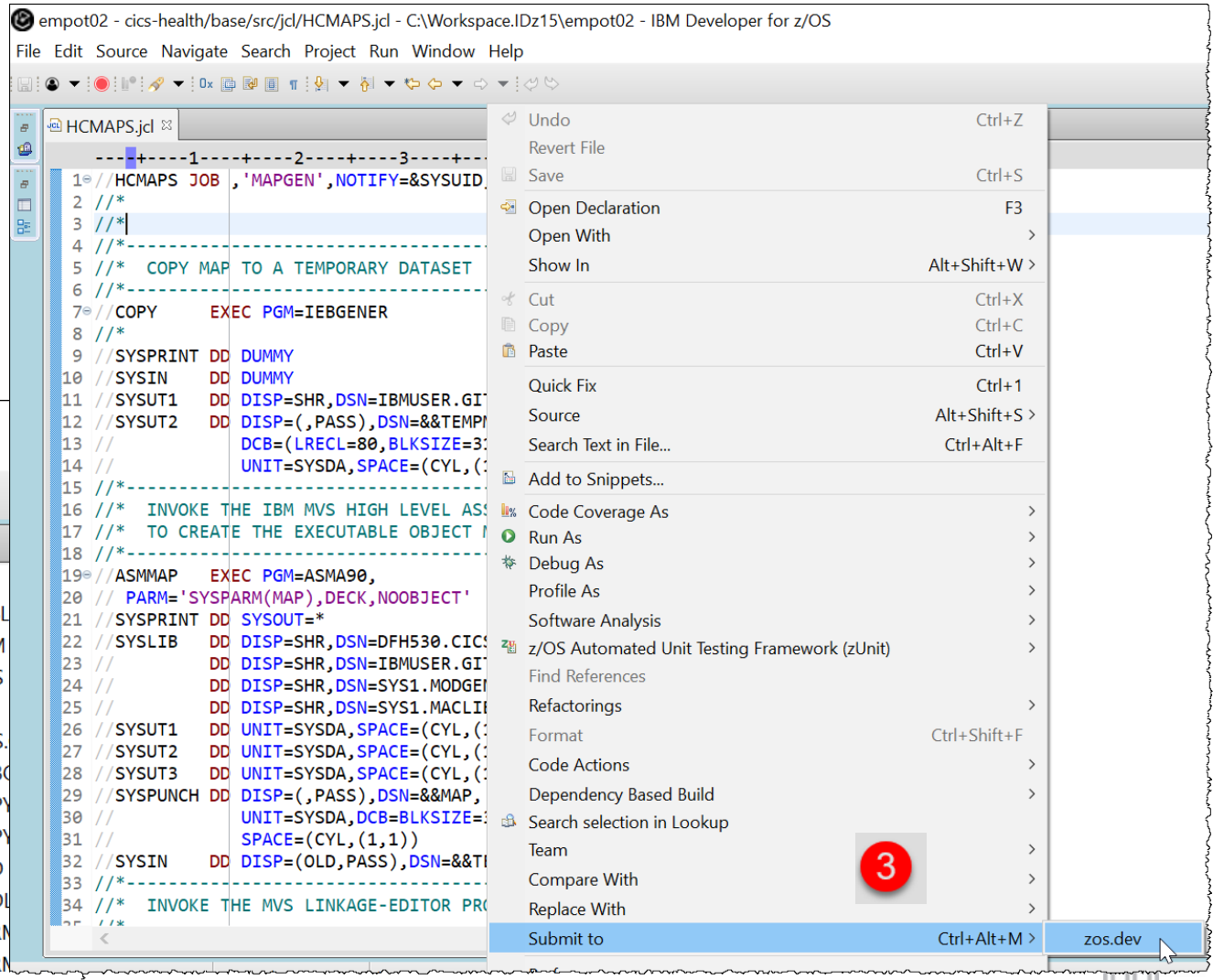
Changing a COBOL/CICS/BMS ”HCMAPS”

1. Source code is loaded at ***LOCAL*** project (example loaded from **Git**)
2. Developer changes the ***CICS BMS*** Map using an editor like **IDz**
3. Developer creates loadlib and copybook on z/OS using **IDz** and **DBB**
   (show using traditional *JCL* versus *DBB User Build*)
4. Developer verifies that the change is successful using CICS
5. Developer ***pushes/commits*** the changes to **Git**
6. **Jenkins** Pipeline starts performing final buildings and deploying to another CICS.
   (Using **zAppBuild** framework and demonstrating the smart building)

Using Traditional JCL versus IBM DBB User Build (example CICS-BMS maps using JCL)







Using Traditional JCL versus IBM DBB User Build (example CICS-BMS maps using JCL)

*HCMAPS.jcl

```
21 //SYSPRINT DD SYSOUT=*
22 //SYSLIB   DD DISP=SHR,DSN=DFH530.CICS.SDFHMAC
23 //          DD DISP=SHR,DSN=IBMUSER.GIT.ZMOBILE.BMS
24 //          DD DISP=SHR,DSN=SYS1.MODGEN
25 //          DD DISP=SHR,DSN=SYS1.MACLIB
26 //SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(10,10))
27 //SYSUT2   DD UNIT=SYSDA,SPACE=(CYL,(10,10))
28 //SYSUT3   DD UNIT=SYSDA,SPACE=(CYL,(10,10))
29 //SYSPUNCH DD DISP=(,PASS),DSN=&&MAP,
30 //            UNIT=SYSDA,DCB=BLKSIZE=3120,
31 //            SPACE=(CYL,(1,1))
32 //SYSIN    DD DISP=(OLD,PASS),DSN=&&TEMPM
33 //*--------------------------------------------------------------
34 //*  INVOKE THE MVS LINKAGE-EDITOR PROGRAM
35 //*--------------------------------------------------------------
36 //LINKMAP  EXEC PGM=IEWL,COND=(7,LT,ASMMAP),
37 //  PARM='LIST,LET,XREF'
38 //*
39 //SYSPRINT DD SYSOUT=*
40 //SYSLMOD  DD DISP=SHR,DSN=IBMUSER.GIT.ZMOBILE.LOAD(HCMAPS)
41 //SYSUT1   DD UNIT=SYSDA,DCB=BLKSIZE=1024,
42 //            SPACE=(CYL,(1,1))
43 //SYSLIN   DD DISP=(OLD,DELETE),DSN=&&MAP
```

IBMUSER.HCMAPS.JOB00136.D0000...

```
58    SYMTRACE
59    TERM                NO
60    TRAP                ON
61    UID                 UNSPECIFIED
62    UPCASE              NO
63    WKSPACE             000000K,000000K
64    XCAL                NO
65    XREF                YES
66    ***END OF OPTIONS***
67
68
69
70
71 SAVE OPERATION SUMMARY:
72
73    MEMBER NAME         HCMAPS
74    LOAD LIBRARY        IBMUSER.GIT.ZMOBILE.LOAD
75    PROGRAM TYPE        PROGRAM OBJECT(FORMAT 2)
76    VOLUME SERIAL       USER05
77    DISPOSITION         REPLACED
78    TIME OF SAVE        12.32.46  MAR  9, 2021
79
80
81 SAVE MODULE ATTRIBUTES:
```

4

```
BROWSE        IBMUSER.GIT.ZMOBILE.LOAD
Command ===>
          Name     Prompt          Alias-of
          HCMAPS                            0
          HCMRESTW                          0
```

*HCMAPS.jcl

```
36 //LINKMAP  EXEC PGM=IEWL,COND=(7,LT,ASMMAP),
37 //  PARM='LIST,LET,XREF'
38 //*
39 //SYSPRINT DD SYSOUT=*
40 //SYSLMOD  DD DISP=SHR,DSN=IBMUSER.GIT.ZMOBILE.LOAD(HCMAPS)
41 //SYSUT1   DD UNIT=SYSDA,DCB=BLKSIZE=1024,
42 //            SPACE=(CYL,(1,1))
43 //SYSLIN   DD DISP=(OLD,DELETE),DSN=&&MAP
44 //*--------------------------------------------------------------
45 //*  INVOKE THE IBM MVS HIGH LEVEL ASSEMBLER
46 //*  TO CREATE THE ASSEMBLER DSECT OF THE MAP (SYSPARM(DSECT))
47 //*  REQUIRED, AT ASSEMBLY TIME, BY PROGRAMS THAT USE THE MAP
48 //*--------------------------------------------------------------
49 //ASMDSECT EXEC PGM=ASMA90,
50 //  PARM='SYSPARM(DSECT),DECK,NOOBJECT'
51 //*
52 //SYSPRINT DD SYSOUT=*
53 //SYSLIB   DD DISP=SHR,DSN=DFH530.CICS.SDFHMAC
54 //          DD DISP=SHR,DSN=IBMUSER.GIT.ZMOBILE.BMS
55 //          DD DISP=SHR,DSN=SYS1.MODGEN
56 //          DD DISP=SHR,DSN=SYS1.MACLIB
57 //SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(10,10))
58 //SYSUT2   DD UNIT=SYSDA,SPACE=(CYL,(10,10))
59 //SYSUT3   DD UNIT=SYSDA,SPACE=(CYL,(10,10))
60 //SYSPUNCH DD DISP=SHR,DSN=IBMUSER.GIT.ZMOBILE.COPYBOOK(HCMAPS)
61 //SYSIN    DD DISP=(OLD,DELETE),DSN=&&TEMPM
62 //
```

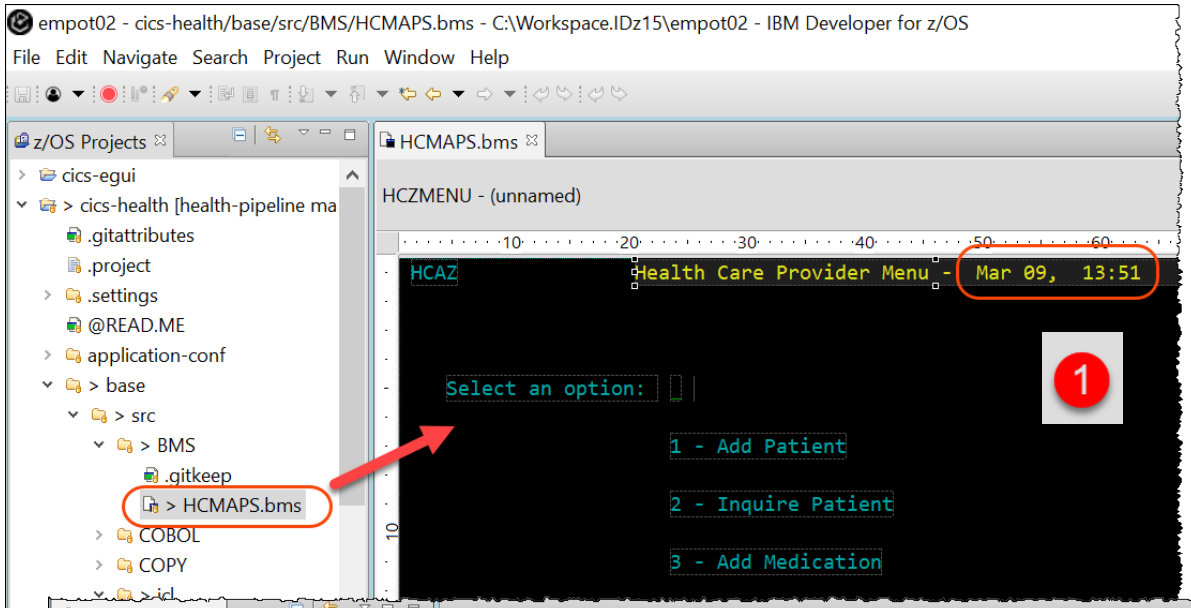
IBMUSER.HCMAPS.JOB00136.D0000103.?.spool

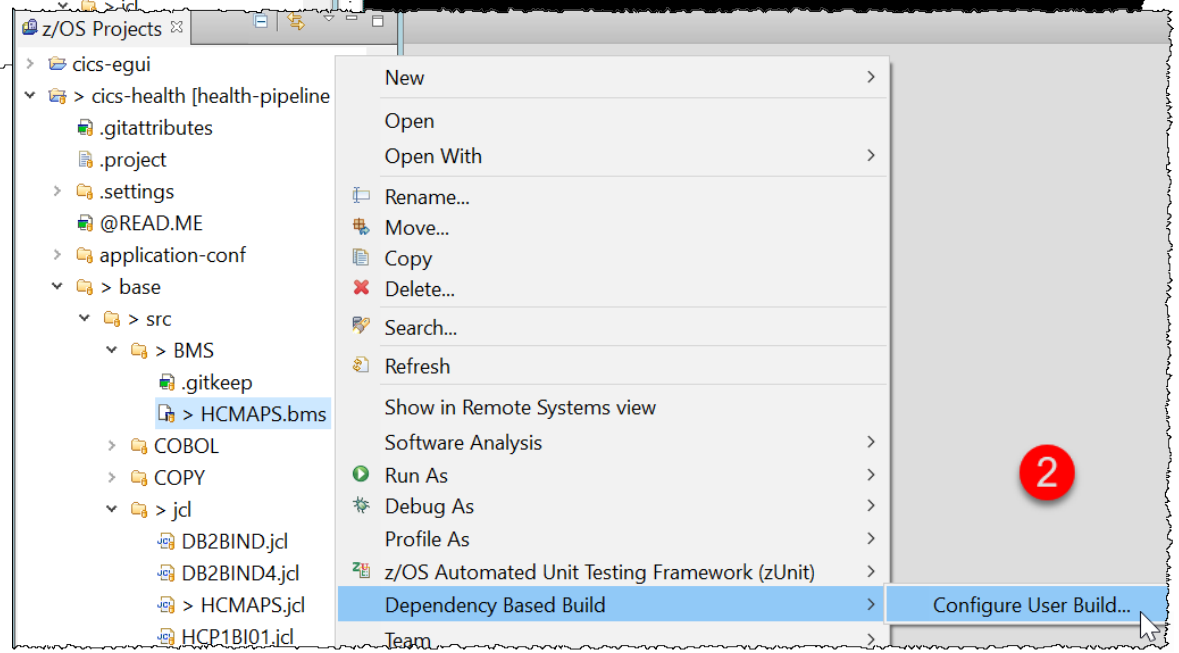
```
1170 L3(D)    (no references identified)
1171 L4(E)    (no references identified)
1172 L5(F)    (no references identified)
1173
1174                              Diagnostic Cross Refe...  Assembler
1175
1176  No Statements Flagged in this Assembly
1177 H LEVEL ASSEMBLER, 5696-234, RELEASE 6.0, PTF UI39092
1178 TEM: z/OS 02.02.00              JOBNAME: HCMAPS      STEPNAME: ASMDSECT
1179 :ode Module:   ASMA047C   From Page     1148   To Page     17584
1180 a Sets Allocated for this Assembly
1181 ı DDname   Data Set Name                                Volume  Member
1182 L SYSIN    SYS21068.T123242.RA000.HCMAPS.TEMPM.H01
1183 L SYSLIB   DFH530.CICS.SDFHMAC                         C2C531
1184 2          IBMUSER.GIT.ZMOBILE.BMS                    USER05
1185 3          SYS1.MODGEN                                C2RES1
1186 4          SYS1.MACLIB                                C2RES1
1187  SYSPRINT IBMUSER.HCMAPS.JOB00136.D0000103.?
1188  SYSPUNCH IBMUSER.GIT.ZMOBILE.COPYBOOK              USER05  HCMAPS
1189
1190 5876K allocated to Buffer Pool       Storage required    2512K
1191  500 Primary Input Records Read         6911 Library Records Read
1192    0 ASMAOPT Records Read               1195 Primary Print Records Writ
1193  500 Object Records Written              0 ADATA Records Written
1194 embly Start Time: 12.32.48 Stop Time: 12.32.49 Processor Time: 00.00.01
1195 urn Code 000
1196
```

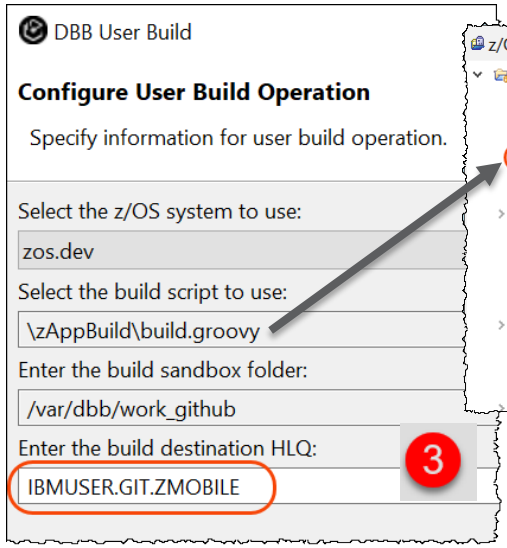
5

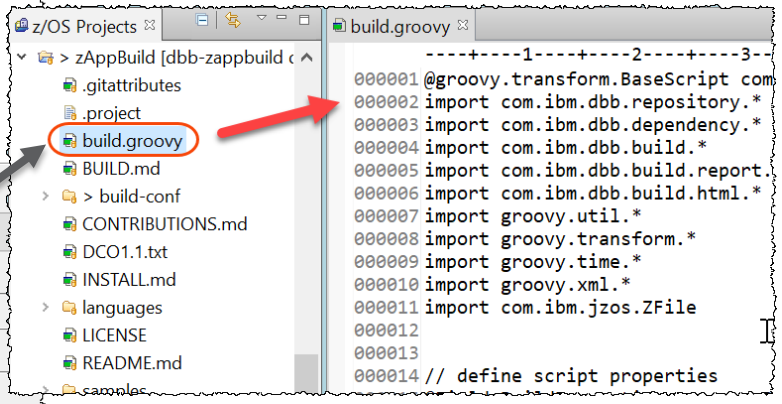
```
BROWSE        IBMUSER.GIT.ZMOBILE.COPYBOOK
Command ===>
          Name     Prompt          Size     Created
          HCCMAREA
          HCCMARE2
          HCERRSPD
          HCERRSWS
          HCMAPS
```

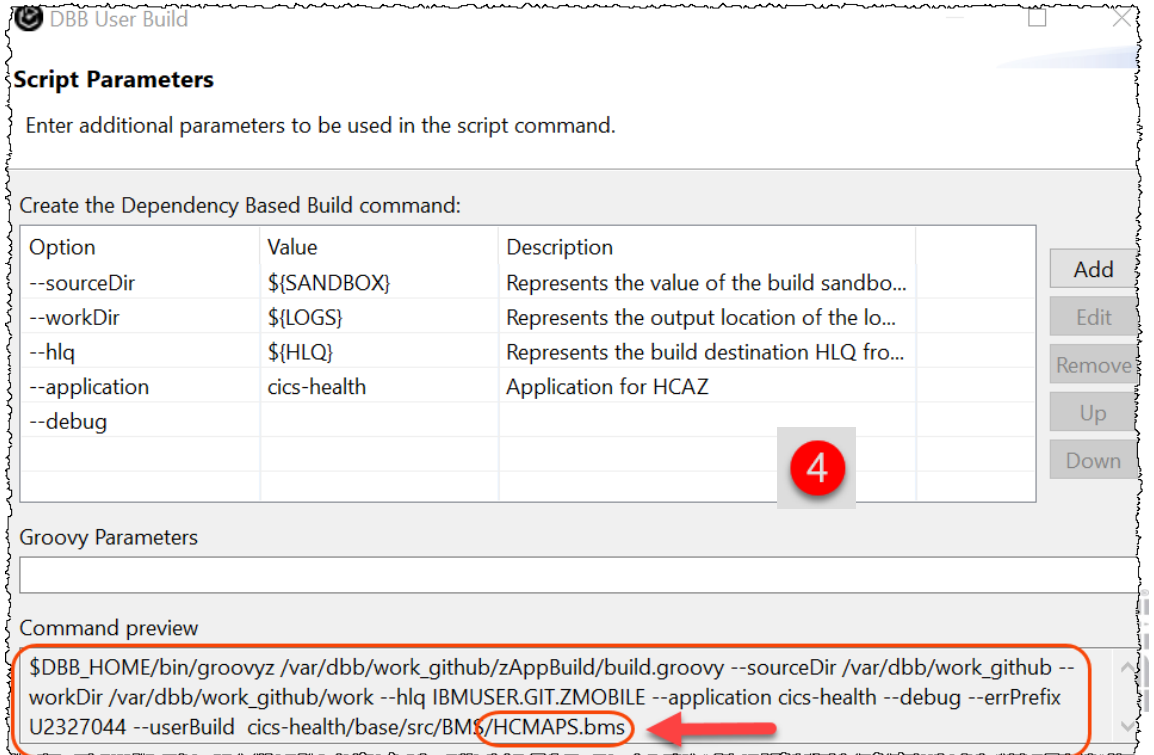
Using Traditional JCL versus IBM DBB User Build (example CICS-BMS maps using DBB)











Using Traditional JCL versus IBM DBB User Build (example CICS-BMS maps using DBB)

```
** Building files mapped to BMS.groovy script
required props = bms_srcPDS,bms_cpyPDS,bms_loadPDS, bms_assembler,bms_linkEditor,

** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.BMS
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.BMS.COPY
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.LOAD
/S0W1/var/dbb/work_github>
*** Building file cics-health/base/src/BMS/HCMAPS.bms

/S0W1/var/dbb/work_github>
** Writing build report data to /var/dbb/work_github/work/BuildReport.json

/S0W1/var/dbb/work_github>
** Writing build report to /var/dbb/work_github/work/BuildReport.html

/S0W1/var/dbb/work_github>
** Build ended at Tue Mar 09 13:16:50 CST 2021

** Build State : CLEAN
** Total files processed : 1
** Total build time  : 35.309 seconds

/S0W1/var/dbb/work_github>
```

5

```
BROWSE          IBMUSER.GIT.ZMOBILE.BMS.COPY
Command ===>
          Name     Prompt      Size    Created
          HCMAPS                7
          **End**
```

DBB User Build Information

DBB User Build is complete for HCMAPS.BMS.

6

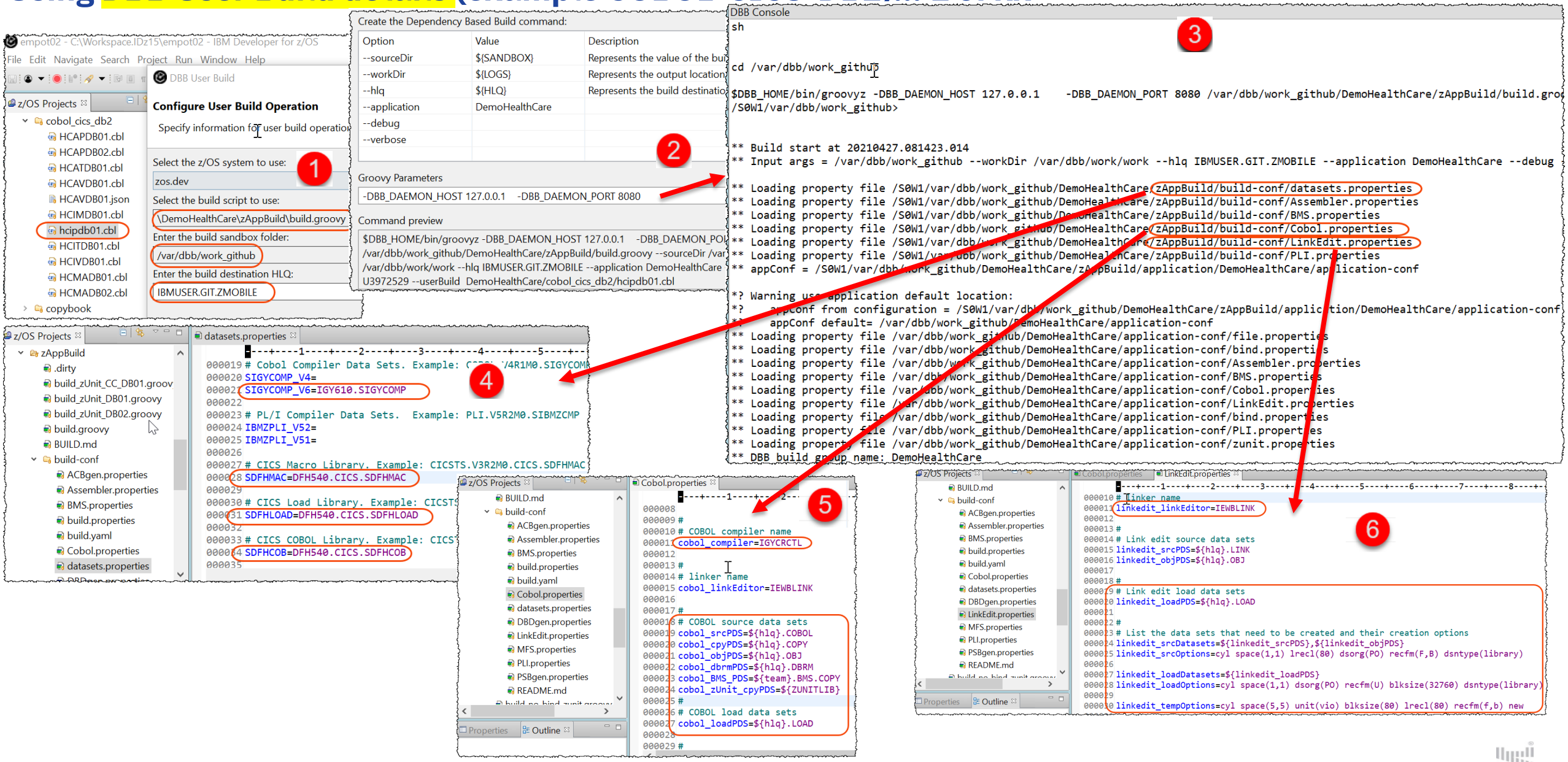
Click Details to see the complete build log.

OK | << Details | Save Log... | Show Build Report

```
1SAVE OPERATION SUMMARY:

  MEMBER NAME      HCMAPS
  LOAD LIBRARY     IBMUSER.GIT.ZMOBILE.LOAD
  PROGRAM TYPE     PROGRAM OBJECT(FORMAT 5 OS COMPAT LEVEL z/OS V1R8 )
  VOLUME SERIAL    USER05
  DISPOSITION      REPLACED
  TIME OF SAVE     13.06.10  MAR  9, 2021
```

```
BROWSE          IBMUSER.GIT.ZMOBILE.LOAD
Command ===>
          Name     Prompt      Alias-of
          HCMAPS                8
          HCMRESTW
```

Using DBB User Build details (example COBOL-CICS-DB2)... 1 of xx



Using DBB User Build details (example COBOL-CICS-DB2)... 2 of xx

DBB Console

```
sh


cd /var/dbb/work_github

$DBB_HOME/bin/groovyz -DBB_DAEMON_HOST 127.0.0.1    -DBB_DAEMON_PORT 8080 /var/dbb/work_github/DemoHealthCare/zAppBuild/buil
/S0W1/var/dbb/work_github>


** Build start at 20210427.081423.014
** Input args = /var/dbb/work_github --workDir /var/dbb/work/work --hlq IBMUSER.GIT.ZMOBILE --application DemoHealthCare --

** Loading property file /S0W1/var/dbb/work_github/DemoHealthCare/zAppBuild/build-conf/datasets.properties
** Loading property file /S0W1/var/dbb/work_github/DemoHealthCare/zAppBuild/build-conf/Assembler.properties
** Loading property file /S0W1/var/dbb/work_github/DemoHealthCare/zAppBuild/build-conf/BMS.properties
** Loading property file /S0W1/var/dbb/work_github/DemoHealthCare/zAppBuild/build-conf/Cobol.properties
** Loading property file /S0W1/var/dbb/work_github/DemoHealthCare/zAppBuild/build-conf/LinkEdit.properties
** Loading property file /S0W1/var/dbb/work_github/DemoHealthCare/zAppBuild/build-conf/PLI.properties
**  appConf = /S0W1/var/dbb/work_github/DemoHealthCare/zAppBuild/application/DemoHealthCare/application-conf

*? Warning use application default location:
*?    appConf from configuration = /S0W1/var/dbb/work_github/DemoHealthCare/zAppBuild/application/DemoHealthCare/application
*?    appConf default= /var/dbb/work_github/DemoHealthCare/application-conf
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/file.properties
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/bind.properties
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/Assembler.properties
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/BMS.properties
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/Cobol.properties
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/LinkEdit.properties
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/bind.properties
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/PLI.properties
** Loading property file /var/dbb/work_github/DemoHealthCare/application-conf/zunit.properties
** DBB build group name: DemoHealthCare
```

1

application-conf

- ACBgen.properties
- application.properties
- Assembler.properties
- bind.properties
- BMS.properties
- Cobol.properties
- DBDgen.properties
- file.properties
- LinkEdit.properties
- MFS.properties
- PLI.properties
- PSBgen.properties
- README.md
- zunit.properties

3

bind.properties

```
000010
000011 # Max Bind RC
000012 bind_maxRC=4
000013
000014 # Flag indicating to run DB2 BIND PACKAGE.  Set to 'true' to bind
000015 #bind_performBindPackage=false
000016 bind_performBindPackage=true
000017
000018 # DBB Toolkit conf (configuration) directory location on zFS. Defa
000019 # This directory may have been relocated to a writable disk by you
000020 bind_runIspfConfDir=/var/dbb/v.1.0.6/conf
000021
000022 # Specify the name of the DB2 subsystem
000023 bind_db2Location=DBBG
000024
000025 # Specify the DB2 collection (Package)
000026 bind_collectionID=HCZMSA1
000027
000028 # Specify the owner of the package, if empty user.name will be use
000029 bind_packageOwner=IBMUSER
000030
000031 # Specify the value of the implicit qualifier.
000032 bind_qualifier=HCZMSA1
```

z/OS Projects

application-conf

- ACBgen.properties
- application.properties
- Assembler.properties
- bind.properties
- BMS.properties
- Cobol.properties
- DBDgen.properties
- file.properties
- LinkEdit.properties
- MFS.properties
- PLI.properties
- PSBgen.properties
- README.md
- zunit.properties

4

Cobol.properties

```
000016 cobol_compilerVersion=V6
000017
000018 #
000019 # default COBOL maximum RCs allowed
000020 # can be overridden by file properties
000021 cobol_compileMaxRC=4
000022 cobol_linkEditMaxRC=0
000023
000024 #
000025 # default COBOL compiler parameters
000026 # can be overridden by file properties
000027 cobol_compileParms=LIB
000028 cobol_compileCICSParms=CICS
000029 cobol_compileSQLParms=SQL
000030 cobol_compileErrorPrefixParms=ADATA,EX(ADX(ELAXMGUX))
000031
000032 # Compile Options for IBM Debugger. Assuming to keep Dwa
000033 # If you would like to separate debug info, additional a
000034 cobol_compileDebugParms=TEST
000035
000036 #
000037 # default LinkEdit parameters
```

z/OS Projects

application-conf

- ACBgen.properties
- application.properties
- Assembler.properties
- bind.properties
- BMS.properties
- Cobol.properties
- DBDgen.properties
- file.properties
- LinkEdit.properties
- MFS.properties
- PLI.properties
- PSBgen.properties

2

file.properties

```
000001 # Application script mappings and file property overrides
000002 # Regi May 19 2020
000003 #
000004 # Script mappings for all application programs
000005 dbb.scriptMapping = Assembler.groovy :: **/*.asm
000006 dbb.scriptMapping = BMS.groovy :: **/*.bms
000007 dbb.scriptMapping = MFS.groovy :: **/*.mfs
000008 dbb.scriptMapping = PSBgen.groovy :: **/psb/*.asm
000009 dbb.scriptMapping = DBDgen.groovy :: **/dbd/*.asm
000010 # was dbb.scriptMapping = Cobol.groovy :: **/*.cbl
000011 # I dont want it process the zUNIT generated COBOL under /testcases
000012 dbb.scriptMapping = Cobol.groovy :: */cobol_cics/*.cbl, \
000013                                     */cobol_cics_db2/*.cbl, \
000014                                     */testcases/*.cbl
000015 dbb.scriptMapping = LinkEdit.groovy :: **/*.lnk
```

Using DBB User Build details (example COBOL-CICS-DB2)... 3 of xx

DBB Console

```
dbb.scriptMapping=BMS.groovy::**/*.bms
dbb.scriptMapping=MFS.groovy::**/*.mfs
dbb.scriptMapping=PLI.groovy::**/*.pli
dbb.scriptMapping=LinkEdit.groovy::**/*.lnk

required props = buildOrder,buildListFileExt,languagePropertyQualifiers
** Build output located at /var/dbb/work/work
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.ASM
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.MACRO
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.OBJ
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.DBRM
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.LOAD
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.BMS
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.BMS.COPY
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.LOAD
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.COBOL
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.COPY
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.OBJ
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.DBRM
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.LOAD
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.LINK
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.OBJ
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.LOAD
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.PLI
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.PLI.INCLUDE
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.OBJ
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.DBRM
** Creating / verifying build dataset IBMUSER.GIT.ZMOBILE.LOAD
** Unable to verify collections. No repository client.
** Adding DemoHealthCare/cobol_cics_db2/hcipdb01.cbl to Building build list
** Writing build list file to /var/dbb/work/work/buildList.txt
DemoHealthCare/cobol_cics_db2/hcipdb01.cbl
** Invoking build scripts according to build order: BMS.groovy,Cobol.groovy,LinkEdit.groovy
/S0W1/var/dbb/work_github>
```

1

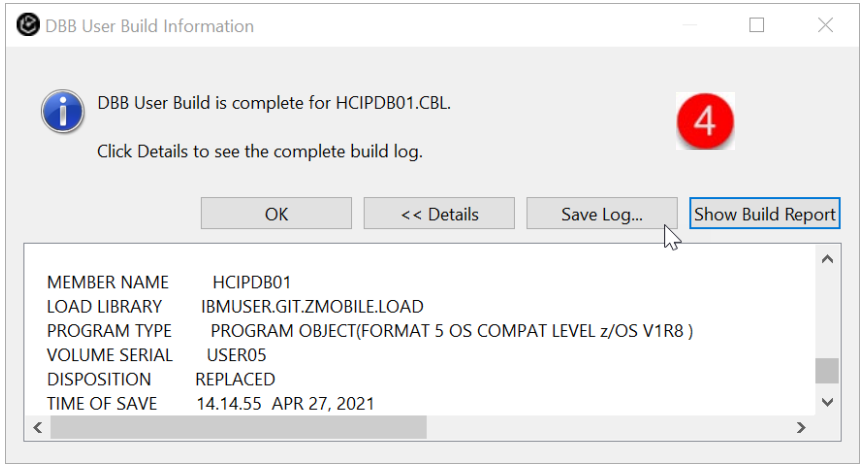
DBB Console

```
** Building files mapped to BMS.groovy script

required props = bms_srcPDS,bms_cpyPDS,bms_loadPDS, bms_assembler,bms_linkEditor,bms_tempOptions,bms_maxRC,   SASMMOD1,SDFHLOAD,SDFHMAC,MACLIB,SCEELKED
/S0W1/var/dbb/work_github>
** Building files mapped to Cobol.groovy script

required props = cobol_srcPDS,cobol_cpyPDS,cobol_objPDS,cobol_loadPDS,cobol_compiler,cobol_linkEditor,cobol_tempOptions,applicationOutputsCollectionName,  SDFHCO

*** Building file DemoHealthCare/cobol_cics_db2/hcipdb01.cbl
*** Creating dependency resolver for DemoHealthCare/cobol_cics_db2/hcipdb01.cbl with [{"library": "SYSLIB", "searchPath": [ {"sourceDir": "/var/dbb/work_github",
/S0W1/var/dbb/work_github>
*** Resolution rules for DemoHealthCare/cobol_cics_db2/hcipdb01.cbl:

{"library":"SYSLIB","searchPath":[{"sourceDir":"\/var\/dbb\/work_github","directory":"DemoHealthCare\/copybook"}]}

*** Physical dependencies for DemoHealthCare/cobol_cics_db2/hcipdb01.cbl:
{"sourceDir":"\/var\/dbb\/work_github","lname":"HCERRSPD","library":"SYSLIB","file":"DemoHealthCare\/copybook\/HCERRSPD.cpy","category":"COPY","resolved":true}
{"sourceDir":"\/var\/dbb\/work_github","lname":"HCERRSWS","library":"SYSLIB","file":"DemoHealthCare\/copybook\/HCERRSWS.cpy","category":"COPY","resolved":true}
{"sourceDir":"\/var\/dbb\/work_github","lname":"HCCMAREA","library":"SYSLIB","file":"DemoHealthCare\/copybook\/HCCMAREA.cpy","category":"SQL INCLUDE","resolved":
{"sourceDir":"\/var\/dbb\/work_github","lname":"SQLCA","library":"SYSLIB","file":"DemoHealthCare\/copybook\/SQLCA.cpy","category":"SQL INCLUDE","resolved":true}

Cobol compiler parms for DemoHealthCare/cobol_cics_db2/hcipdb01.cbl = LIB,CICS,SQL,ADATA,EX(ADX(ELAXMGUX)),TEST

Copying /var/dbb/work/work/linkCard.lnk to IBMUSER.GIT.ZMOBILE.LINK(HCIPDB01)
/S0W1/var/dbb/work_github>
*** Binding DemoHealthCare/cobol_cics_db2/hcipdb01.cbl

*** Copying /var/dbb/work/work/bind.clist to IBMUSER.GIT.ZMOBILE.DBRM.CLIST(BIND)

*** Executing CLIST to bind program DemoHealthCare/cobol_cics_db2/hcipdb01.cbl
/S0W1/var/dbb/work_github>
** Building files mapped to LinkEdit.groovy script

required props = linkedit_srcPDS,linkedit_objPDS,linkedit_loadPDS,linkedit_linkEditor,linkedit_tempOptions,applicationOutputsCollectionName,  SDFHLOAD,SCEELKED
```

2

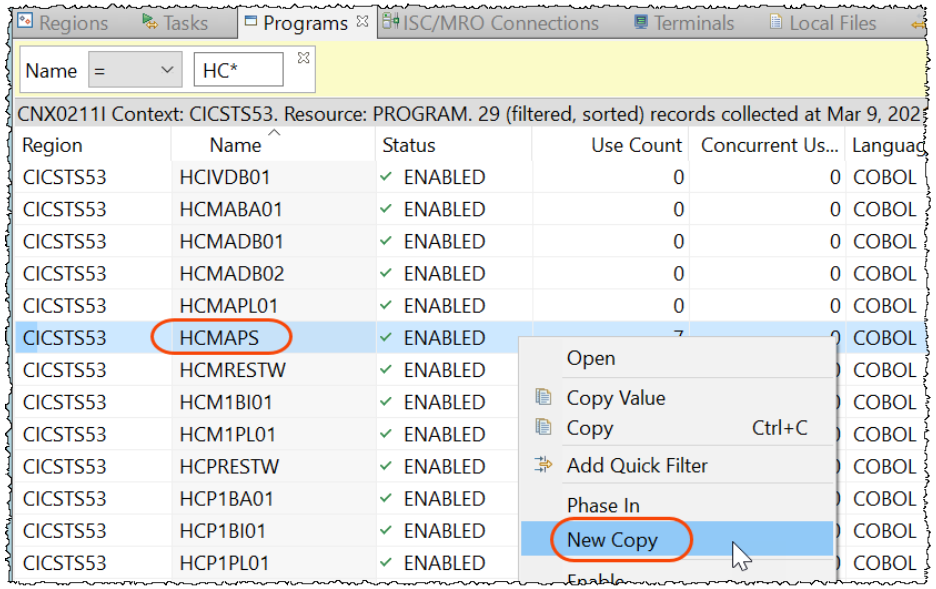
```
** Writing build report data to /var/dbb/work/work/BuildReport.json
** Writing build report to /var/dbb/work/work/BuildReport.html
** Build ended at Tue Apr 27 20:15:19 GMT 2021
** Build State : CLEAN
** Total files processed : 1
** Total build time  : 56.201 seconds

__RC=0__
** Build finished
/S0W1/var/dbb/work_github>
```

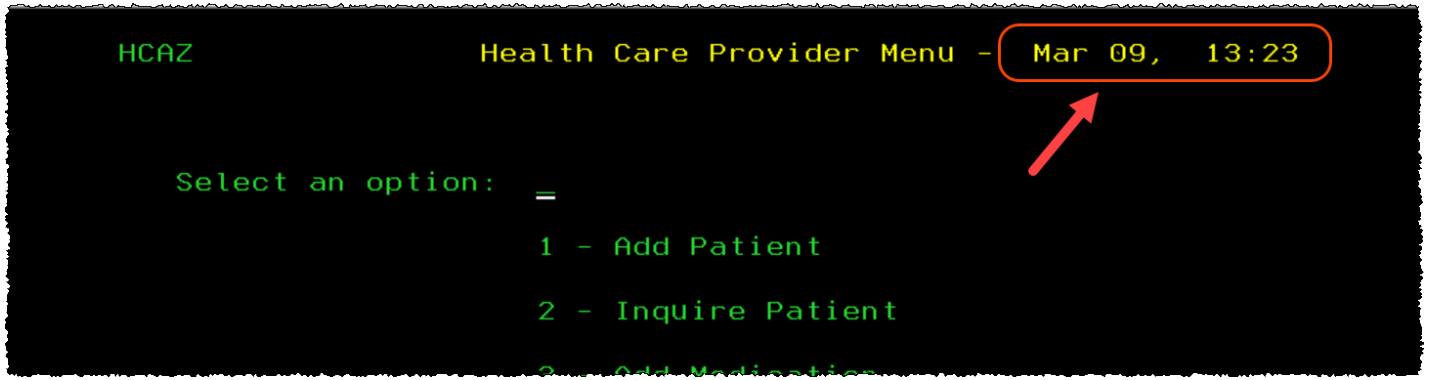
3



4

Using CICS verify the change (HCAZ transaction)

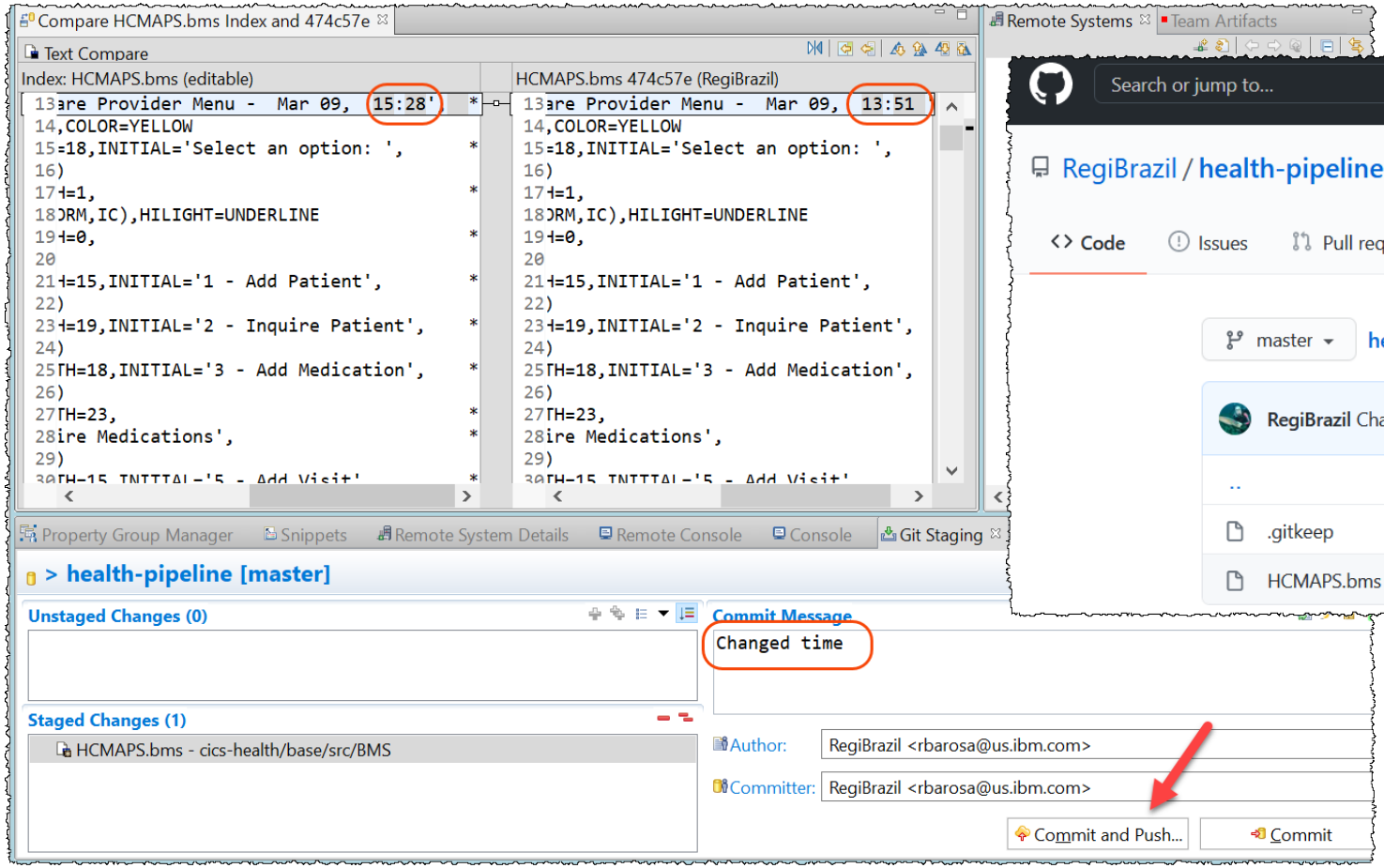


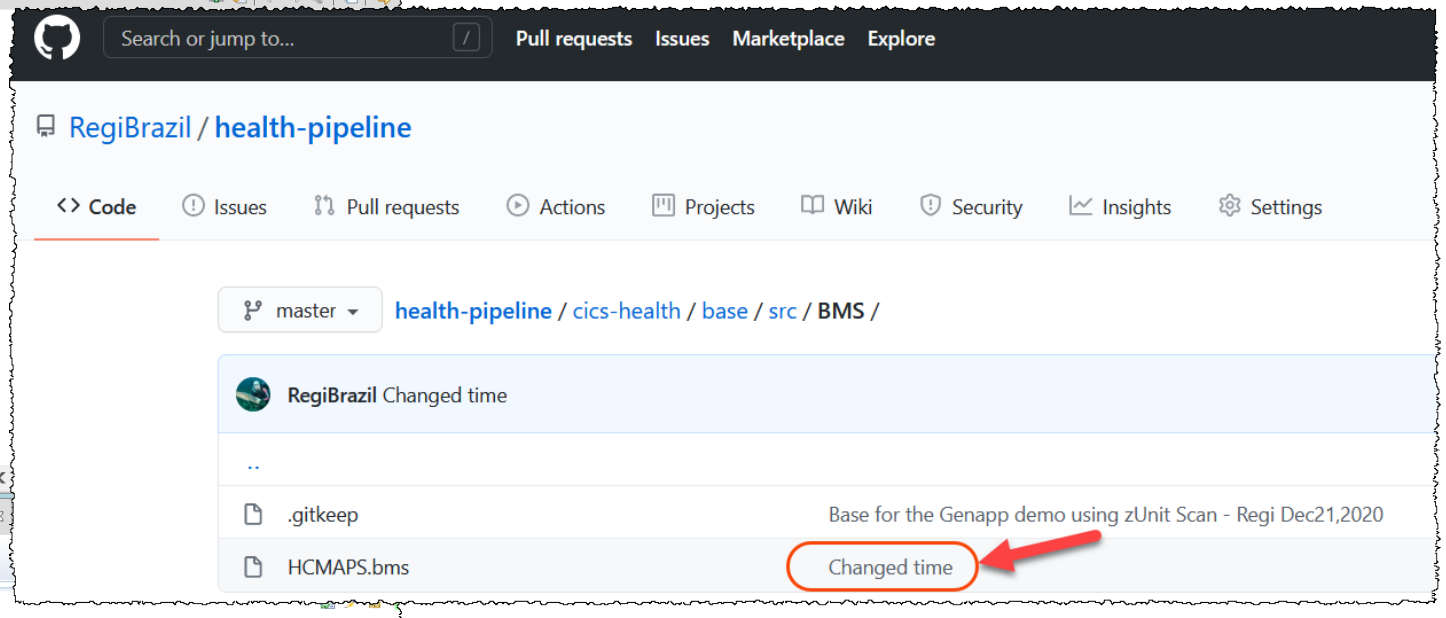


Changing a COBOL/CICS/BMS ”HCMAPS”

1. Source code is loaded at ***LOCAL*** project (example loaded from **Git**)
2. Developer changes the ***CICS BMS*** Map using an editor like **IDz**
3. Developer creates loadlib and copybook on z/OS using **IDz** and **DBB**
   (show using traditional *JCL* versus *DBB User Build*)
4. Developer verifies that the change is successful using CICS
5. Developer ***pushes/commits*** the changes to **Git**
6. **Jenkins** Pipeline starts performing final buildings and deploying to another CICS.
   (Using **zAppBuild** framework and demonstrating the smart building)

Developer pushes/commits the changes to Git

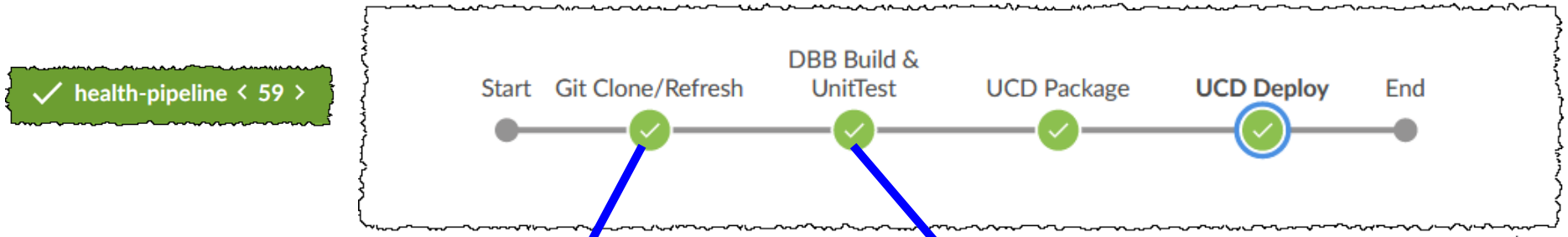




Changing a COBOL/CICS/BMS ”HCMAPS”

1. Source code is loaded at ***LOCAL*** project (example loaded from **Git**)
2. Developer changes the ***CICS BMS*** Map using an editor like **IDz**
3. Developer creates loadlib and copybook on z/OS using **IDz** and **DBB**
   (show using traditional *JCL* versus *DBB User Build*)
4. Developer verifies that the change is successful using CICS
5. Developer ***pushes/commits*** the changes to **Git**
6. **Jenkins** Pipeline starts performing final buildings and deploying to another CICS.
   (Using zAppBuild framework and demonstrating the smart building)

Jenkins Pipeline starts performing final buildings and deploying to another CICS



Git Clone/Refresh - 1m 17s

- Clone from GitHub to Rocket Git on zOS — Print Message
- rm -f .git/info/sparse-checkout — Shell Script
- Check out from version control

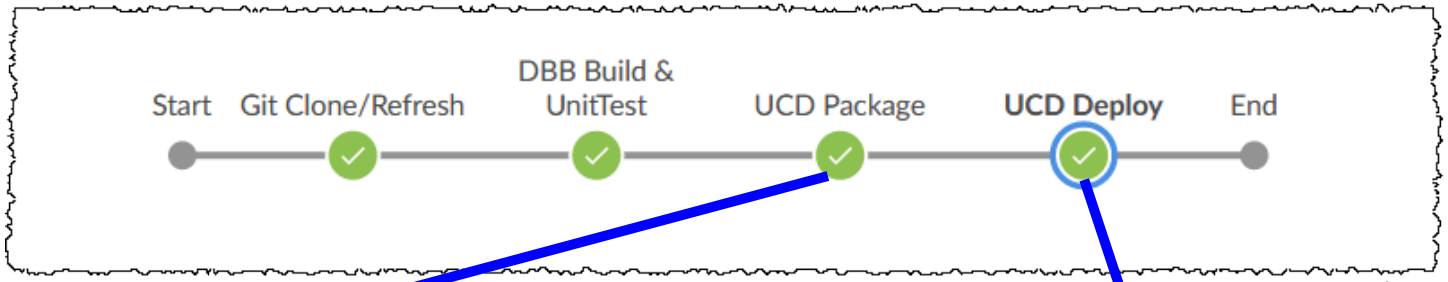
```
The recommended git tool is: NONE
No credentials specified
Fetching changes from the remote Git repository
 > /var/dbb/1.1/bin/git-jenkins2.sh rev-parse --is-inside-work-tree # timeout
 > /var/dbb/1.1/bin/git-jenkins2.sh config remote.origin.url git@github.com:R
Fetching upstream changes from git@github.com:RegiBrazil/health-pipeline.git
 > /var/dbb/1.1/bin/git-jenkins2.sh --version # timeout=10
 > git --version # 'git version 2.14.4_zos_b09'
 > /var/dbb/1.1/bin/git-jenkins2.sh fetch --tags --progress -- git@github.com
Checking out Revision ba8a59fa95f77b2022f434e5506fa2d8a6a33d77 (origin/master
 > /var/dbb/1.1/bin/git-jenkins2.sh rev-parse origin/master^{commit} # timeou
 > /var/dbb/1.1/bin/git-jenkins2.sh config core.sparsecheckout # timeout=10
 > /var/dbb/1.1/bin/git-jenkins2.sh read-tree -mu HEAD # timeout=10
 > /var/dbb/1.1/bin/git-jenkins2.sh checkout -f ba8a59fa95f77b2022f434e5506fa
Commit message: "Changed time"
 > /var/dbb/1.1/bin/git-jenkins2.sh rev-list --no-walk 1ea4474af48f96e190ac32
```

Shell Script

```
+ /var/dbb/1.1/bin/groovyz /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/dbb-zappbuild/build.groovy --logEncoding UTF-8 -w /var/jenkin
--application cics-health --sourceDir /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/cics-health --workDir /var/jenkins/agent/e2e-pipel
IBMUSER.DBB.HEALTH --url https://clmweb:11043/dbb/ -pw ADMIN -i
Cannot contact e2e-pipeline: java.lang.InterruptedException

** Build start at 20210309.083602.036
** Repository client created for https://clmweb:11043/dbb/
** Build output located at /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-59/build.20210309.083602.036
** Build result created for BuildGroup:cics-health-master BuildLabel:build.20210309.083602.036 at https://clmweb:11043/dbb/rest/buildResult/11741
** --impactBuild option selected. Building impacted programs for application cics-health
** Writing build list file to /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-59/build.20210309.083602.036/buildList.txt
** Invoking build scripts according to build order: BMS.groovy,Cobol.groovy
** Invoking test scripts according to test order: ZunitConfig.groovy
** Building files mapped to BMS.groovy script
*** Building file cics-health/base/src/BMS/HCMAPS.bms
** Building files mapped to Cobol.groovy script
*** Building file cics-health/base/src/COBOL/HCM1PL01.cbl
*** Building file cics-health/base/src/COBOL/HCT1PL01.cbl
*** Building file cics-health/base/src/COBOL/HCV1PL01.cbl
*** Building file cics-health/base/src/COBOL/HCMAPL01.cbl
*** Building file cics-health/base/src/COBOL/HCAZMENU.cbl
*** Building file cics-health/base/src/COBOL/HCP1PL01.cbl
** Writing build report data to /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-59/build.20210309.083602.036/BuildReport.json
** Writing build report to /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-59/build.20210309.083602.036/BuildReport.html
** Build ended at Tue Mar 09 20:38:41 GMT 2021
** Build State : CLEAN
** Total files processed : 7
** Total build time  : 2 minutes, 39.096 seconds

** Build finished
```

This is a “smart building”

Jenkins Pipeline starts performing final buildings and deploying to another CICS



Shell Script

```
+ /var/dbb/1.1/bin/groovyz /var/jenkins/agent/e2e-pipeline/work
HC_CICS --workDir /var/jenkins/agent/e2e-pipeline/workspace/he
** Create version start at 20210309.083958.039
** Properties at startup:
   component -> HC_CICS
   startTime -> 20210309.083958.039
   workDir -> /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-59
   preview -> false
   buztoolPath -> /apps/ucd/v7/bin/buztool.sh
** Read build report data from /var/jenkins/agent/e2e-pipeline/workspace/health-
** Find deployable outputs in the build report
   ! Buildrecord type TYPE_COPY_TO_PDS is supported with DBB toolkit 1.0.8 and h
   IBMUSER.DBB.HEALTH.LOAD(HCMAPS), MAPLOAD
   IBMUSER.DBB.HEALTH.LOAD(HCM1PL01), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCT1PL01), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCV1PL01), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCMAPL01), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCAZMENU), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCP1PL01), LOAD
** Generate UCD ship list file
** Write ship list file to  /var/jenkins/agent/e2e-pipeline/workspace/health-pip
** Following UCD buztool cmd will be invoked
/apps/ucd/v7/bin/buztool.sh createzosversion -c HC_CICS -s /var/jenkins/agent/e
/workspace/health-pipeline/BUILD-59/build.20210309.083602.036/buztool.output
** Create version by running UCD buztool
zOS toolkit config   : /apps/ucd/v7/ (7.0.2.0,20190131-0837)
zOS toolkit binary   : /apps/ucd/v7/ (7.0.2.0,20190131-0837)
zOS toolkit data set : BUZV7 (7.0.2.0,20190131-0837)
Reading parameters:
....Command : createzosversion
....Component : HC_CICS
....Generate version name : 20210309-204111
....Shiplist file : /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-59/build.20
....Output File:/var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-59/build.202103
Verifying version
....Repository location : /apps/ucd/v7/var/repository/HC_CICS/20210309-204111
Pre-processing shiplist:
```

Publish Artifacts to IBM UrbanCode Deploy

```
Deploying component versions '{HC_CICS=[latest]}'
Starting deployment process 'Deploy HC to CICS + MF to Windows' of application 'A HC zOS COBOL CICS' in environment 'TEST'
Deployment request id is: '17818b91-1dd2-e4ad-daf8-2959ab9d9603'
Deployment is running. Waiting for UCD Server feedback.
```

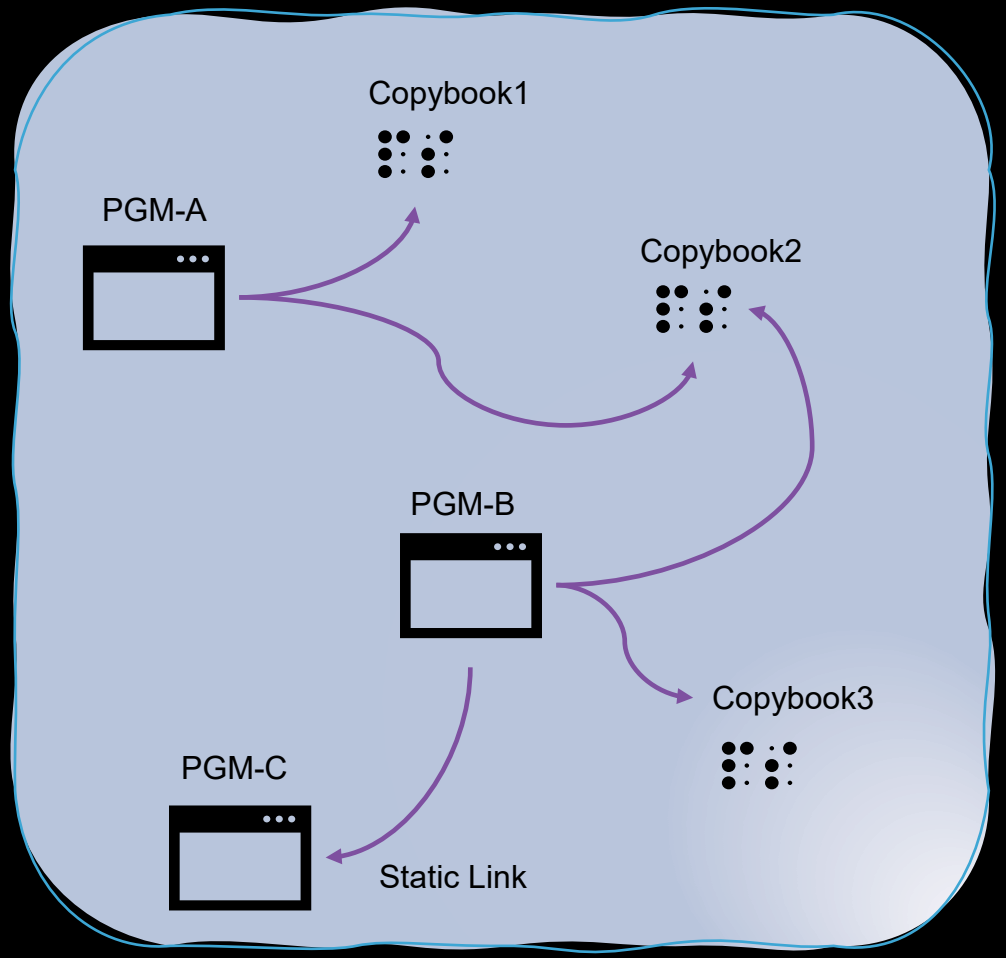
3. Install HC_CICS	1 / 1	3:41:26 PM	0:07:17	Success
HC_CICS	1 / 1	3:41:26 PM	0:07:17	Success
Deploy HC to CICS (HC_CICS 20210309-204111)		3:41:26 PM	0:07:17	Success
1. Download Artifacts for zOS		3:41:28 PM	0:01:31	Success
2. Deploy Data Sets for CICS		3:42:59 PM	0:01:44	Success
3. GenBndCard		3:44:43 PM	0:01:23	Success
4. Generate Program List		3:44:43 PM	0:01:23	Success
5. bindPackage & PLAN		3:46:06 PM	0:01:46	Success
6. NEWCOPY Programs		3:47:52 PM	0:00:51	Success
Total Execution	1 / 1	3:41:26 PM	0:07:17	Success

```
2021/03/09 20:48:55.704 GMT BUZCP0006I Connected to "10.1.1.2:1491". CICS TS ver
2021/03/09 20:49:03.639 GMT BUZCP0037I Perform NEWCOPY Operation.
2021/03/09 20:49:04.382 GMT BUZCP0024I NEWCOPY PROGRAM "HCAZMENU" succeeded.
2021/03/09 20:49:04.871 GMT BUZCP0024I NEWCOPY PROGRAM "HCM1PL01" succeeded.
2021/03/09 20:49:05.115 GMT BUZCP0024I NEWCOPY PROGRAM "HCMAPL01" succeeded.
2021/03/09 20:49:05.343 GMT BUZCP0024I NEWCOPY PROGRAM "HCMAPS" succeeded.
2021/03/09 20:49:05.750 GMT BUZCP0024I NEWCOPY PROGRAM "HCP1PL01" succeeded.
2021/03/09 20:49:06.127 GMT BUZCP0024I NEWCOPY PROGRAM "HCT1PL01" succeeded.
2021/03/09 20:49:06.532 GMT BUZCP0024I NEWCOPY PROGRAM "HCV1PL01" succeeded.
2021/03/09 20:49:06.684 GMT BUZCP0029I Summary: 7 NEWCOPY request(s) succeeded,
```

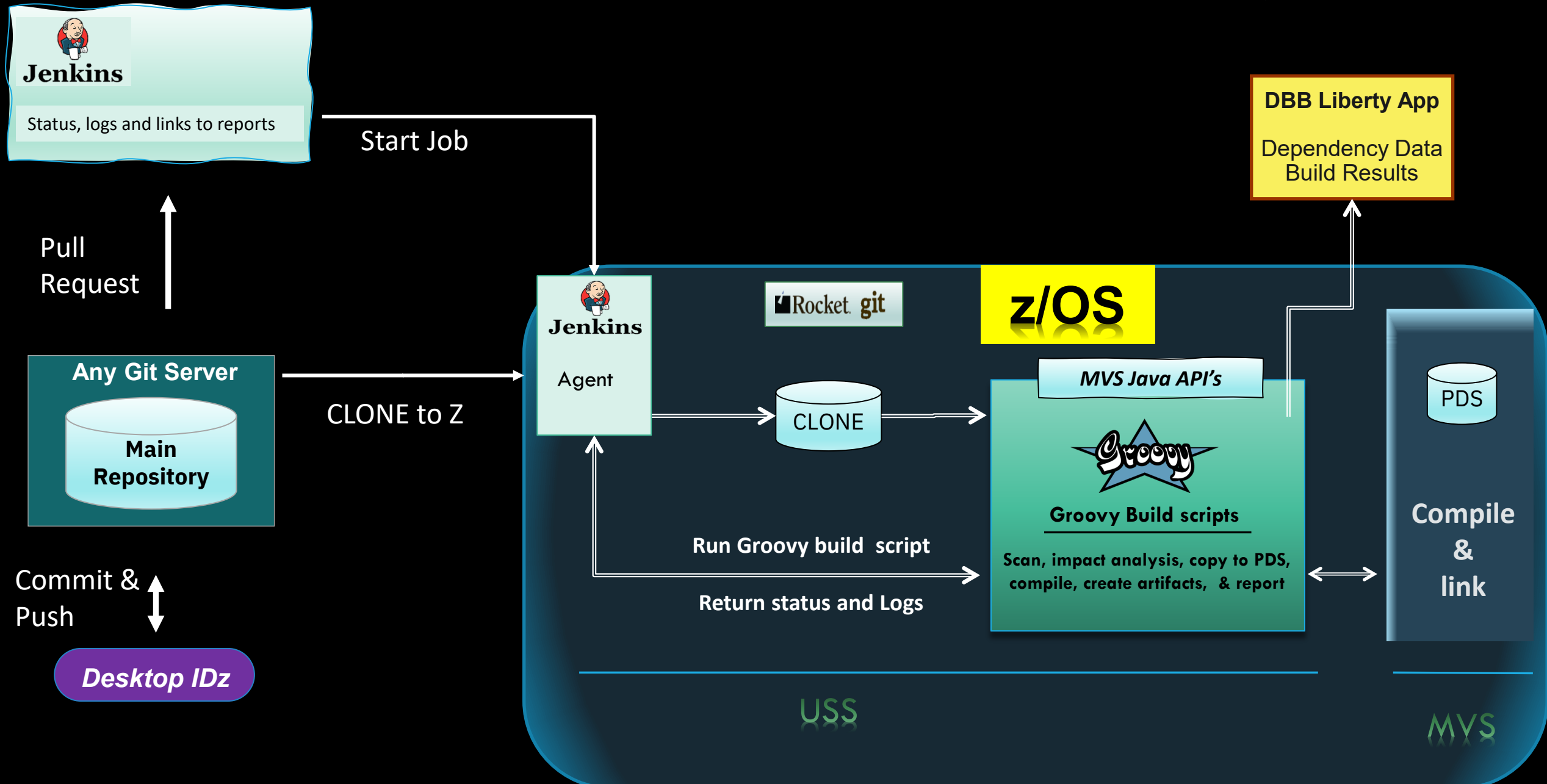
DBB's Intelligent Build

Example:
Dependencies between programs & copybooks are used to drive intelligent, efficient builds

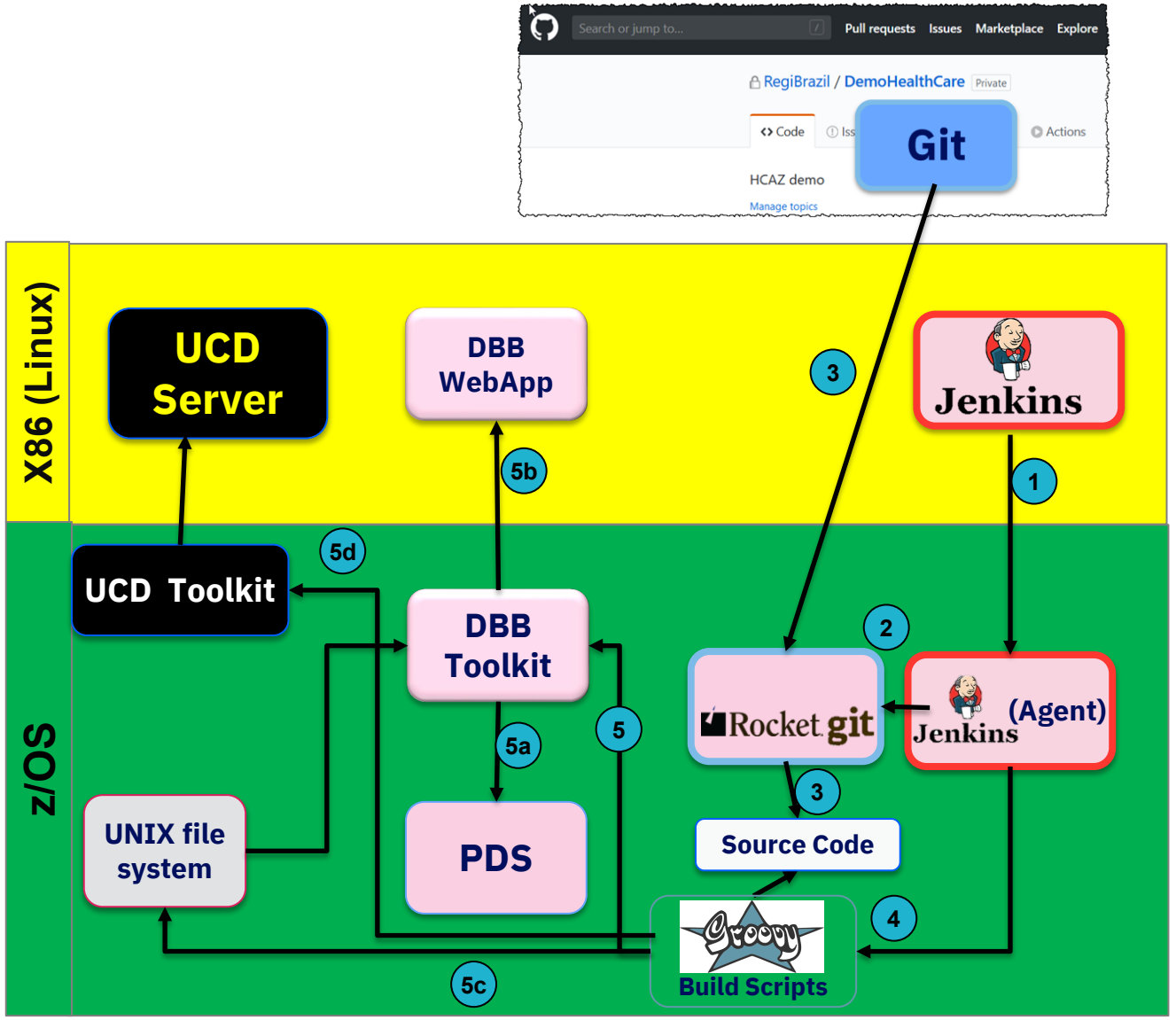
Also can identify zUnit tests to be executed on builds



DBB Architecture (Continuous Integration Phase)



High Level Demo Diagram using **Jenkins-Git-DBB-UCD**(UrbanCode Deploy)



[1] **Jenkins** server sends build commands to zOS remote agent.

[2] **Jenkins** agent issues **Git pull** command to update **zOS Rocket Git** repository (get latest updates).

[3] **zOS Rocket's Git** client automatically converts source from **UTF-8** to **EBCDIC** during pull.

[4] **Jenkins** agent invokes build scripts containing **DBB** APIs using zOS Git repository (Rocket).

[5] **DBB Toolkit** provides Java APIs to:
[5a] Create datasets, copy source from **zFS** to **PDS**, invoke zOS Compiler/Linkage Editor.

[5b] **DBB** scan and store dependency data from source files, perform dependency and impact analysis, store build results on DBB WebApp .

[5c] Copy logs from **PDS** to **zFS**, generate build reports (can be saved in build result).

[5d] Invoke UCD toolkit to create a deployable version at UCD Server (Linux)

DBB Groovy Starter Kit – aka zAppBuild

- zAppBuild is set of sample Groovy scripts used to run a DBB Build (Compile and Link)
- Its generic and customizable
- builds any z/OS application (COBOL, Assembler, PL1)
- delivered with the product or from our Public GitHub Site
- Supports IDz User Builds and full Pipeline Builds
- **Main Purpose -** *accelerate* client adoption of DBB
- https://github.com/ibm/dbb

What is the zAppBuild Framework

- A generic build solution for building z/OS applications
  - Uses Apache Groovy build scripts and IBM Dependency Based Build (DBB) APIs.
  - Contains high-level enterprise-wide settings
  - Enables use of application-level settings to provide necessary setting overrides for each application
- Delivered as a public sample in GitHub Repository
  - https://github.com/IBM/dbb-zappbuild
  - https://github.com/ibm/dbb
- Repository is intended to be cloned to a single location on Unix Systems Services (USS) and used to build all of your z/OS applications.
  - Refer to “Build Configuration” for additional site-specific set-up information
    - https://github.com/IBM/dbb-zappbuild/blob/development/build-conf/README.md

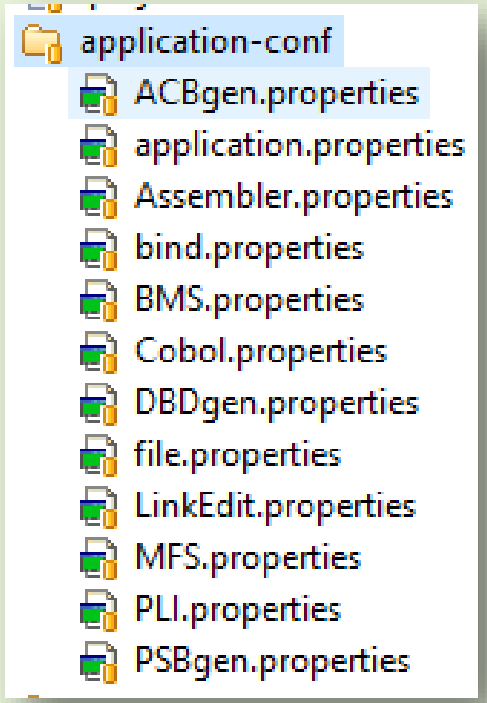
zAppBuild Highlights

- Single Build Script that supports the following languages out of the box:
  - Assembler
  - BMS and MFS
  - COBOL
  - PL/I
  - Link Cards
  - PSB, DBD
- Supported build actions:
  - **Single Program –** Build a single program in the application
  - **List of Programs –** Build a list of programs provided by a text file
  - **Full Build –** Build all programs (or buildable files) of an application
  - **Impact Build –** Build only the programs impacted by source files that have changed since the last successful build
  - **Scan Only –** Skip the actual building and only scan source files for dependency data

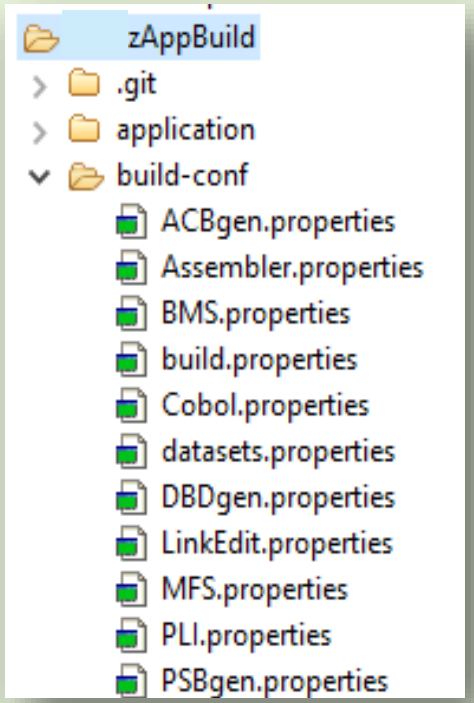
zAppBuild Architecture

Configuration Properties

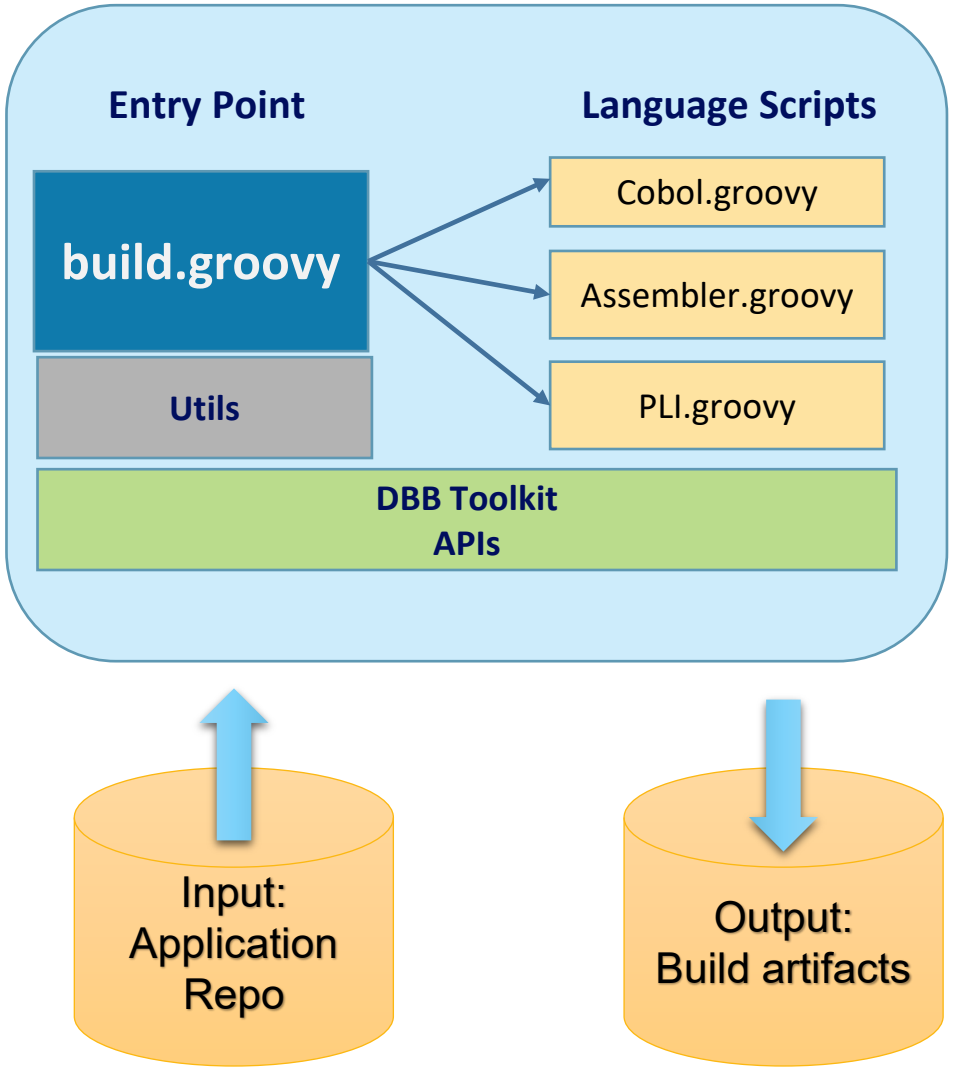
Application level



Enterprise level

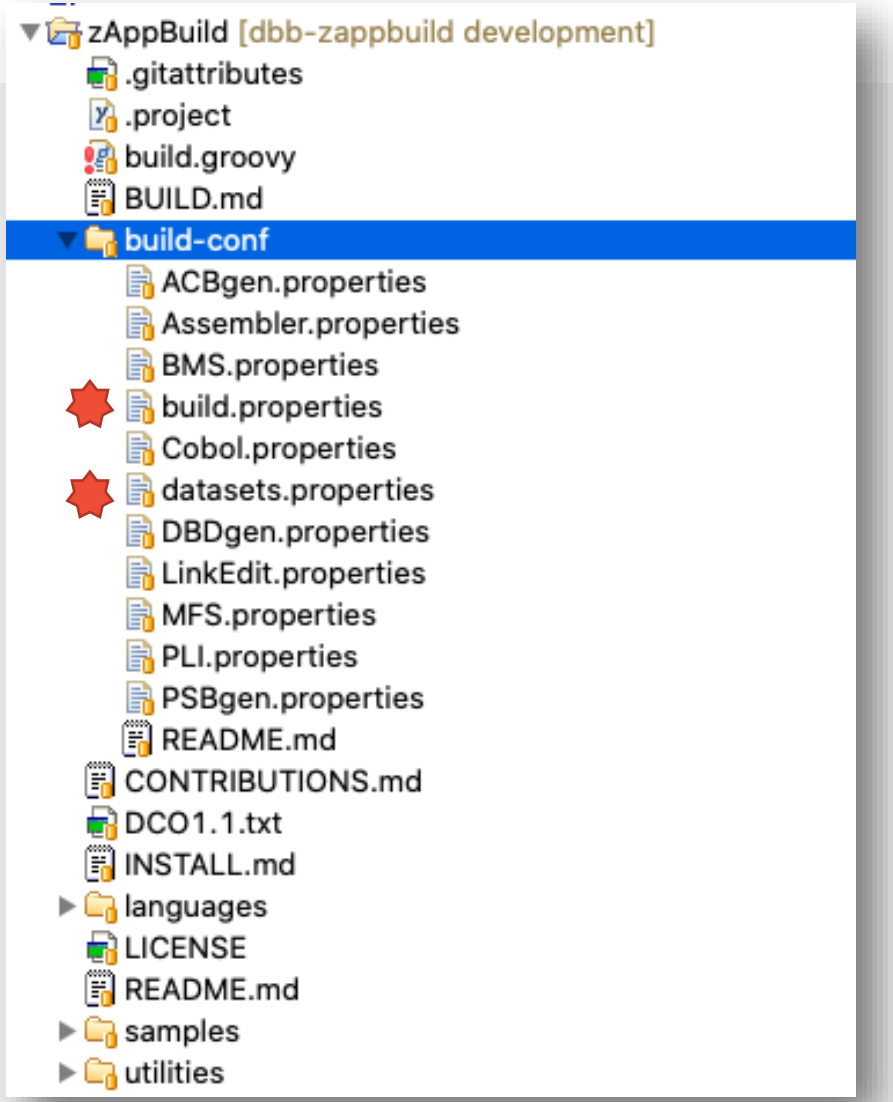


zAppBuild build framework



Customization of the central zAppBuild build framework

- The zAppBuild implementation is configurable through a set of property files.
- It is up to the central build team to define these common settings for the enterprise.
- zAppBuild provides property files in the ***build-conf*** directory
- ✸ Indicates files requiring mandatory customization.



build-conf/build.properties

Build.properties defines

- the references to the additional central property files
- access to the IBM Dependency Based Build web server
- Configuration for the usage of the ISPF client gateway

build.properties

```
# Build properties used by build.groovy

#
# Comma separated list of additional build property files to load
# Supports both relative path (to zAppBuild/build-conf/) and absolute path
buildPropFiles=datasets.properties,Assembler.properties,BMS.properties,MFS.properties,PSBgen.properties,DBDg

#
# file extension that indicates the build file is really a build list or build list filter
buildListFileExt=txt

#
# Alternate root directory for application-conf locations.  Allows for the deployment of
# the application-conf directories to an alternate location rather in the application repository.
# The expectation is that the root directory will have subfolders for all of the applications built
# by zAppBuild in which the actual application-conf directory is located:
#
# Example: Static location on USS
# applicationConfRootDir=/u/build/config/applications
# |- /u/build/config/applications
# |                       |- App1
# |                       |    |- application-conf
# |                       |    |        |- application.properties
# |                       |- App2
# |                       |    |- application-conf
#
# Example: Application cofig files stored in zAppBuild
# applicationConfRootDir=${zAppBuildDir}/applications
# |- /u/build/zAppBuild/applications
# |                           |- App1
# |                           |    |- application-conf
# |                           |    |        |- application.properties
# |                           |- App2
```

build-conf/dataset.properties

- The dataset.properties file defines the client specific locations of Language Environment, Compilers, Subsystem Libraries like IMS, DB2 or CICS.

datasets.properties

```
# Dataset references
# Build properties for Partition Data Sets (PDS) used by zAppBuild build scripts.
# Please provide a fully qualified DSN for each build property below.
# Ex:
# MACLIB=SYS1.MACLIB

# z/OS macro library. Example: SYS1.MACLIB
MACLIB=

# Assembler macro library. Example: CEE.SCEEMAC
SCEEMAC=

# LE (Language Environment) load library. Example: CEE.SCEELKED
SCEELKED=

# High Level Assembler (HLASM) load library. Example: ASM.SASMMOD1
SASMMOD1=

# Cobol Compiler Data Sets. Example: COBOL.V4R1M0.SIGYCOMP
SIGYCOMP_V4=
SIGYCOMP_V6=

# PL/I Compiler Data Sets.  Example: PLI.V5R2M0.SIBMZCMP
IBMZPLI_V52=
IBMZPLI_V51=

# CICS Macro Library. Example: CICSTS.V3R2M0.CICS.SDFHMAC
SDFHMAC=

# CICS Load Library. Example: CICSTS.V3R2M0.CICS.SDFHLOAD
SDFHLOAD=

# CICS COBOL Library. Example: CICSTS.V3R2M0.CICS.SDFHCOB
```

build-conf/<language>.properties

- The central team defines the common preferences for all language scripts.
  Primarily these are:
  - Dataset naming conventions for source and target datasets
  - Dataset characteristics.

Attention: The default value might not be sufficient for an enterprise rollout.

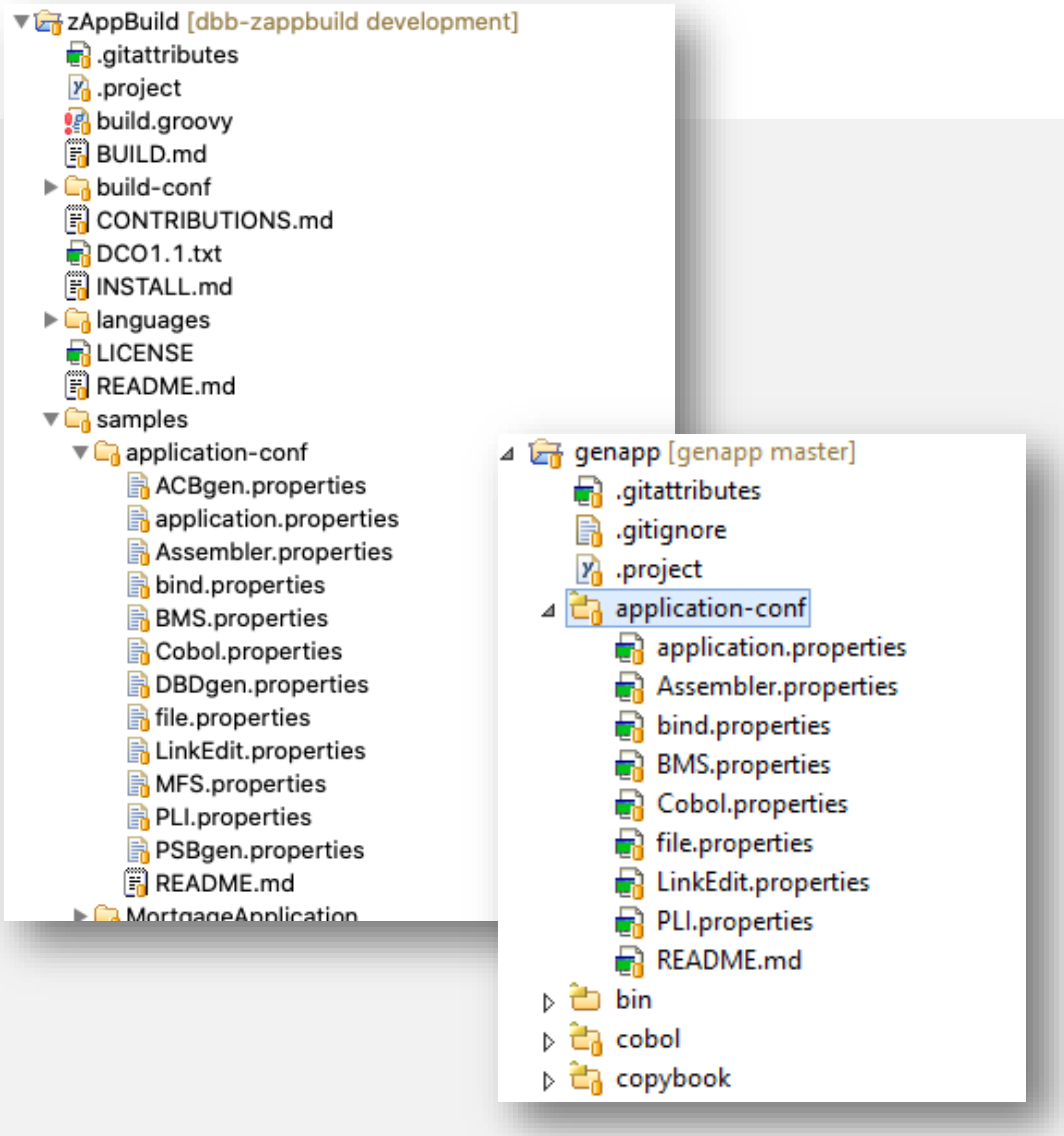
These files do not require customization.

build.properties | Cobol.properties

```
# Releng properties used by language/Cobol.groovy

#
# Comma separated list of required build properties for Cobol.groovy
cobol_requiredBuildProperties=cobol_srcPDS,cobol_cpyPDS,cobol_objPDS,cobol_loadPDS,\
  cobol_compiler,cobol_linkEditor,cobol_tempOptions,applicationOutputsCollectionName,\
  SDFHCOB,SDFHLOAD,SDSNLOAD,SCEELKED

#
# COBOL compiler name
cobol_compiler=IGYCRCTL

#
# linker name
cobol_linkEditor=IEWBLINK

#
# COBOL source data sets
cobol_srcPDS=${hlq}.COBOL
cobol_cpyPDS=${hlq}.COPY
cobol_objPDS=${hlq}.OBJ
cobol_dbrmPDS=${hlq}.DBRM
cobol_BMS_PDS=${team}.BMS.COPY
#
# COBOL load data sets
cobol_loadPDS=${hlq}.LOAD

#
# List the data sets that need to be created and their creation options
cobol_srcDatasets=${cobol_srcPDS},${cobol_cpyPDS},${cobol_objPDS},${cobol_dbrmPDS}
cobol_srcOptions=cyl space(1,1) lrecl(80) dsorg(PO) recfm(F,B) dsntype(library)

cobol_loadDatasets=${cobol_loadPDS}
```

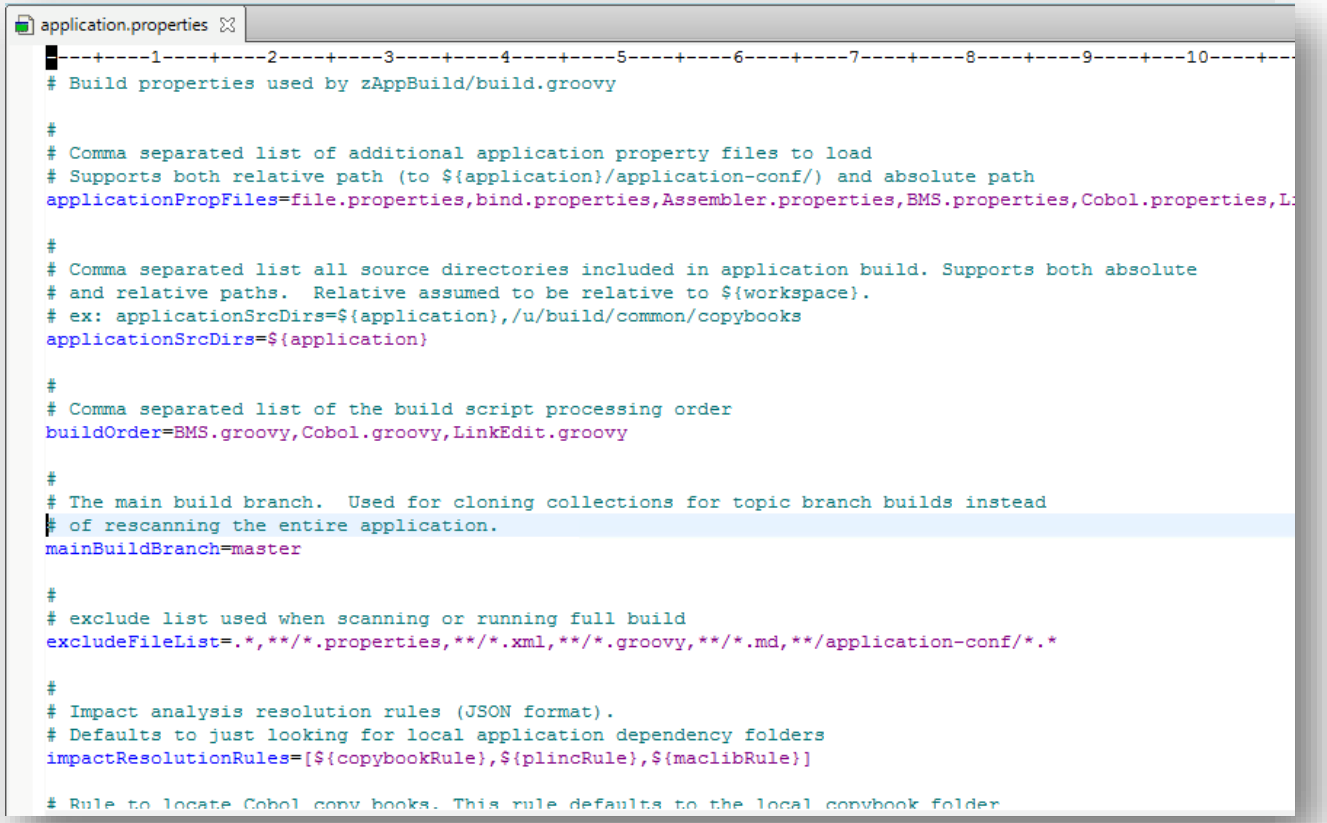
Application level customizations

- Each application repository (build scope) requires an ***application-conf*** directory which defines the application specific build configuration properties.
- zAppBuild ships a template in ***samples/application-conf***, which needs to be copied to each application repository and then customized to the application needs.



application-conf/application.properties

- Application.properties is a key configuration file managed by the application team.
- It defines the
  - **buildOrder** – the language scripts used for a certain application,
  - **applicationSrcDirs** - which need to be considered in the build
  - excluded files in the **excludeFileList**
  - searchPaths for Dependency Analysis and Impact Analysis.



application-conf/file.properties

- Is the central configuration file to map files in the repository to the available zAppBuid build scripts.
- It is also used to to overwrite specific build properties.
  - For example fileBuildRanks to determine the order of processing for a certain languagescript
  - Overwriting file characteristics.

file.properties

```
# Application script mappings and file property overrides

#
# Script mappings for all application programs
dbb.scriptMapping = Assembler.groovy :: **/*.asm
dbb.scriptMapping = BMS.groovy :: **/*.bms
dbb.scriptMapping = MFS.groovy :: **/*.mfs
dbb.scriptMapping = PSBgen.groovy :: **/psb/*.asm
dbb.scriptMapping = DBDgen.groovy :: **/dbd/*.asm
dbb.scriptMapping = Cobol.groovy :: **/*.cbl
dbb.scriptMapping = LinkEdit.groovy :: **/*.lnk
dbb.scriptMapping = PLI.groovy :: **/*.pli
```

file.properties | file.properties

```
# Application script mappings and file property overrides

#
# Script mappings for all application programs
dbb.scriptMapping = Assembler.groovy :: **/*.asm
dbb.scriptMapping = BMS.groovy :: **/*.bms
dbb.scriptMapping = Cobol.groovy :: **/*.cbl
dbb.scriptMapping = LinkEdit.groovy :: **/*.lnk
dbb.scriptMapping = PLI.groovy :: **/*.pli

#
# Need to build epsnbrvl.cbl first during cobol builds
cobol_fileBuildRank = 1 :: **/cobol/epsnbrvl.cbl

#
# Skip creating a load module for these programs as they will be statically linked to
cobol_linkEdit = false :: **/cobol/epsnbrvl.cbl, **/cobol/epsmlist.cbl

#
# epsmlist needs to compile as CICS but does not have EXEC CICS statements
# so is not automatically flagged as CICS by dependency scanner
isCICS = true :: **/cobol/epsmlist.cbl
```

application-conf/ <language>.properties

file.properties | file.properties | Cobol.properties

```
# Application properties used by zAppBuild/language/Cobol.groovy

#
# default COBOL program build rank - used to sort language build file list
# leave empty - overridden by file properties if sorting needed
cobol_fileBuildRank=

#
# COBOL dependency resolution rules
# Rules defined in application.properties
cobol_resolutionRules=[${copybookRule}]

#
# default COBOL compiler version
# can be overridden by file properties
cobol_compilerVersion=V6

#
# default COBOL maximum RCs allowed
# can be overridden by file properties
cobol_compileMaxRC=4
cobol_linkEditMaxRC=0

#
# default COBOL compiler parameters
# can be overridden by file properties
cobol_compileParms=LIB
cobol_compileCICSParms=CICS
cobol_compileSQLParms=SQL
cobol_compileErrorPrefixParms=ADATA,EX(ADX(ELAXMGUX))

# Compile Options for IBM Debugger. Assuming to keep Dwarf Files inside the load.
# If you would like to separate debug info, additional allocations needed (See COBOL +
cobol_compileDebugParms=TEST

#
# default LinkEdit parameters
# can be overridden by file properties
cobol_linkEditParms=MAP,RENT,COMPAT(PM5)

# If you would like to have a physical link card, we generated it for you given the be
# This property has priority over cobol_linkDebugExit
# cobol_linkEditStream=    INCLUDE OBJECT(@{member})
cobol_linkEditStream=

# If using a debug exit, provide the SYSLIN instream DD
# Samp: cobol_linkDebugExit=    INCLUDE OBJECT(@{member})  \n    INCLUDE SYSLIB(EQAD3C
cobol_linkDebugExit=    INCLUDE OBJECT(@{member})  \n    INCLUDE SYSLIB(EQAD3CXT)


#
# execute link edit step
# can be overridden by file properties
cobol_linkEdit=true

#
# scan link edit load module for link dependencies
# can be overridden by file properties
cobol_scanLoadModule=true
```

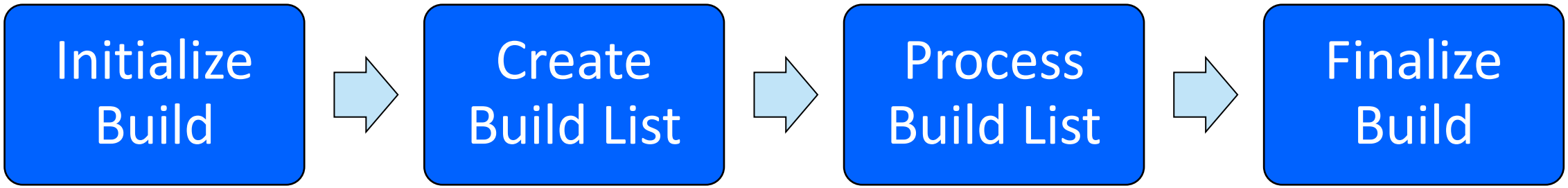
application-conf/ <language>.properties

- These property files contain the application properties used be the language scripts.
- Key elements:
  - Compile / Link Options
  - Load module scanner

file.properties | file.properties | Cobol.properties

```
# Application properties used by zAppBuild/language/Cobol.groovy

#
# default COBOL program build rank - used to sort language build file list
# leave empty - overridden by file properties if sorting needed
cobol_fileBuildRank=

#
# COBOL dependency resolution rules
# Rules defined in application.properties
cobol_resolutionRules=[${copybookRule}]

#
# default COBOL compiler version
# can be overridden by file properties
cobol_compilerVersion=V6

#
# default COBOL maximum RCs allowed
# can be overridden by file properties
cobol_compileMaxRC=4
cobol_linkEditMaxRC=0

#
# default COBOL compiler parameters
# can be overridden by file properties
cobol_compileParms=LIB
cobol_compileCICSParms=CICS
cobol_compileSQLParms=SQL
cobol_compileErrorPrefixParms=ADATA,EX(ADX(ELAXMGUX))

# Compile Options for IBM Debugger. Assuming to keep Dwarf Files inside the load.
# If you would like to separate debug info, additional allocations needed (See COBOL +
cobol_compileDebugParms=TEST

#
# default LinkEdit parameters
# can be overridden by file properties
cobol_linkEditParms=MAP,RENT,COMPAT(PM5)

# If you would like to have a physical link card, we generated it for you given the be
# This property has priority over cobol_linkDebugExit
# cobol_linkEditStream=    INCLUDE OBJECT(@{member})
cobol_linkEditStream=

# If using a debug exit, provide the SYSLIN instream DD
# Samp: cobol_linkDebugExit=    INCLUDE OBJECT(@{member})  \n    INCLUDE SYSLIB(EQAD3C
cobol_linkDebugExit=    INCLUDE OBJECT(@{member})  \n    INCLUDE SYSLIB(EQAD3CXT)


#
# execute link edit step
# can be overridden by file properties
cobol_linkEdit=true

#
# scan link edit load module for link dependencies
# can be overridden by file properties
cobol_scanLoadModule=true
```

High Level View of zAppBuild Workflow



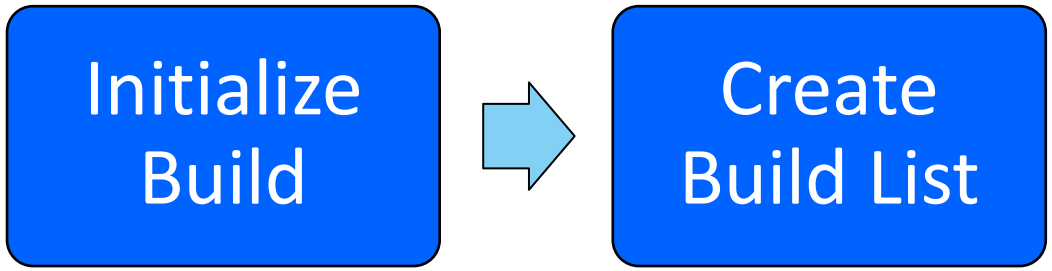
zAppBuild Workflow

Initialize Build

- Load properties
- Initialize Build Report and build result
- Create Data Sets if needed
- Connect to WebApp
- Initialize Collection if needed

Pipeline build only

zAppBuild Workflow



- Load properties
- Initialize Build Report and build result
- Create Data Sets if needed
- Connect to WebApp
- Initialize Collection if needed

- Process build option
- User build, full build, impact, etc.
- Scan changed files and update collection
- Perform impact analysis
- List of files to build

Pipeline build only

zAppBuild Workflow

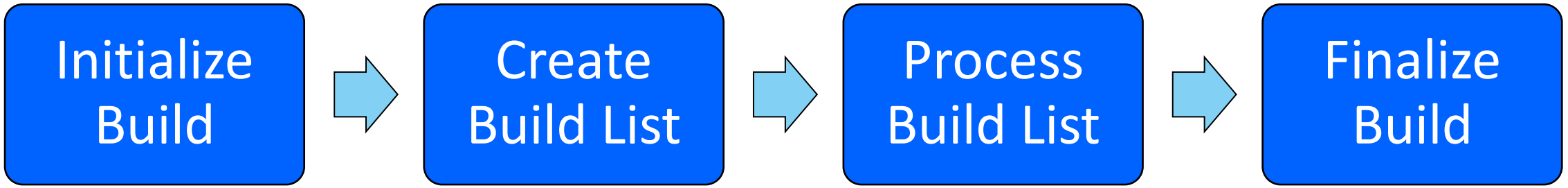
- Load properties
- Initialize Build Report and build result
- Create Data Sets if needed
- Connect to WebApp
- Initialize Collection if needed

- Process build option
- User build, full build, impact, etc.
- Scan changed files and update collection
- Perform impact analysis
- List of files to build

- Find appropriate language script to run
- Run Language script
- Run additional processes (file level)

Pipeline build only

zAppBuild Workflow



- Load properties
- Initialize Build Report and build result
- Create Data Sets if needed
- Connect to WebApp
- Initialize Collection if needed

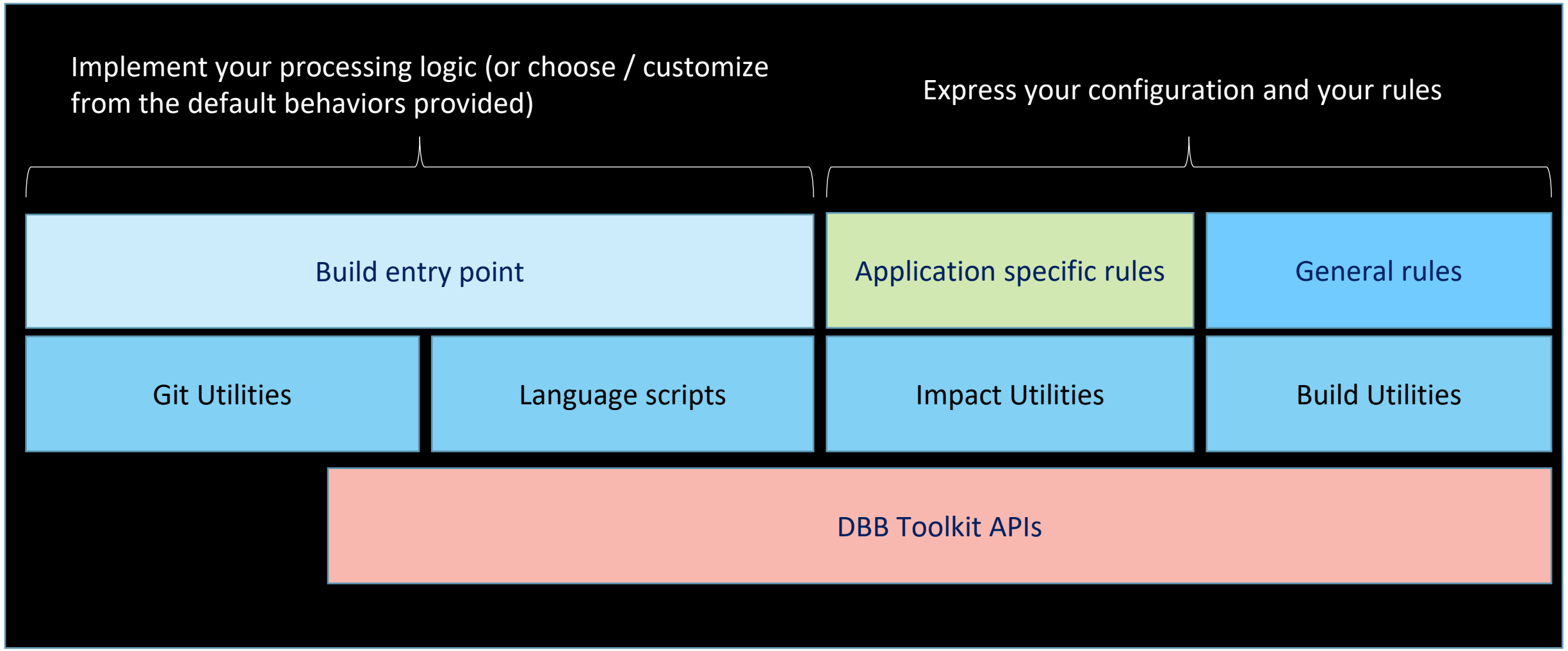
- Process build option
- User build, full build, impact, etc.
- Scan changed files and update collection
- Perform impact analysis
- List of files to build

- Find appropriate language script to run
- Run Language script
- Run additional processes (file level)

- Run additional processes (build level)
- Mark Build as finished with status
- Add git hashes
- Upload logs and reports

Pipeline build only

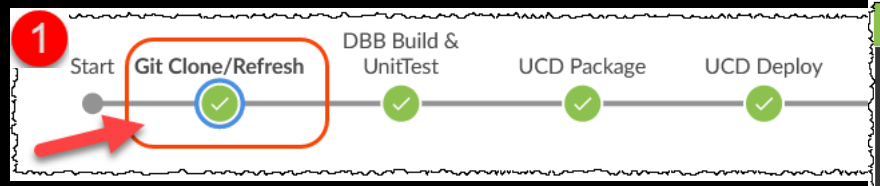
zAppBuild Building Blocks

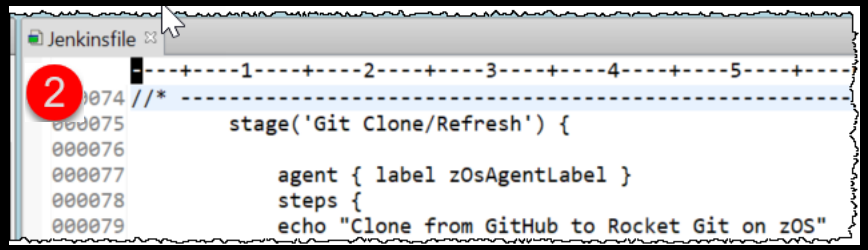


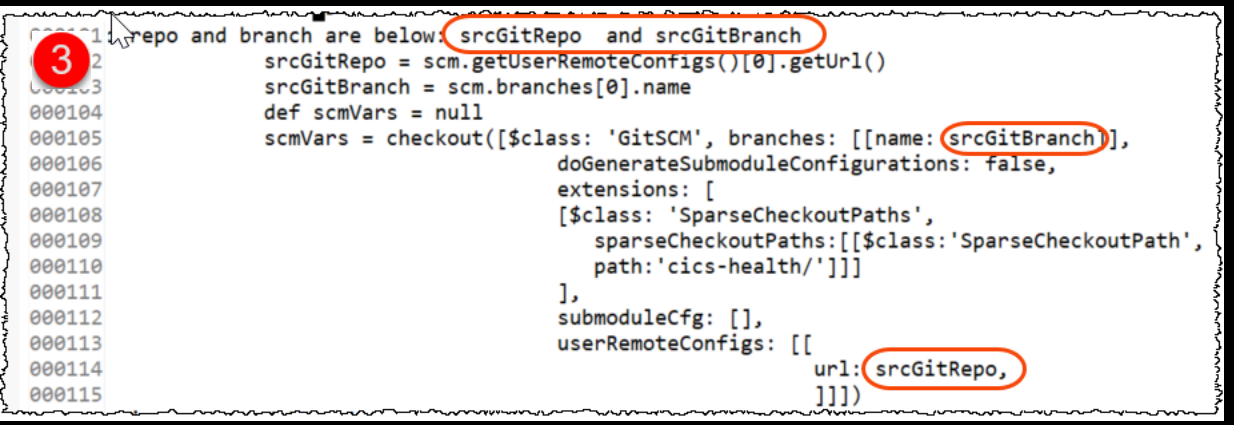
zAppBuild Summary

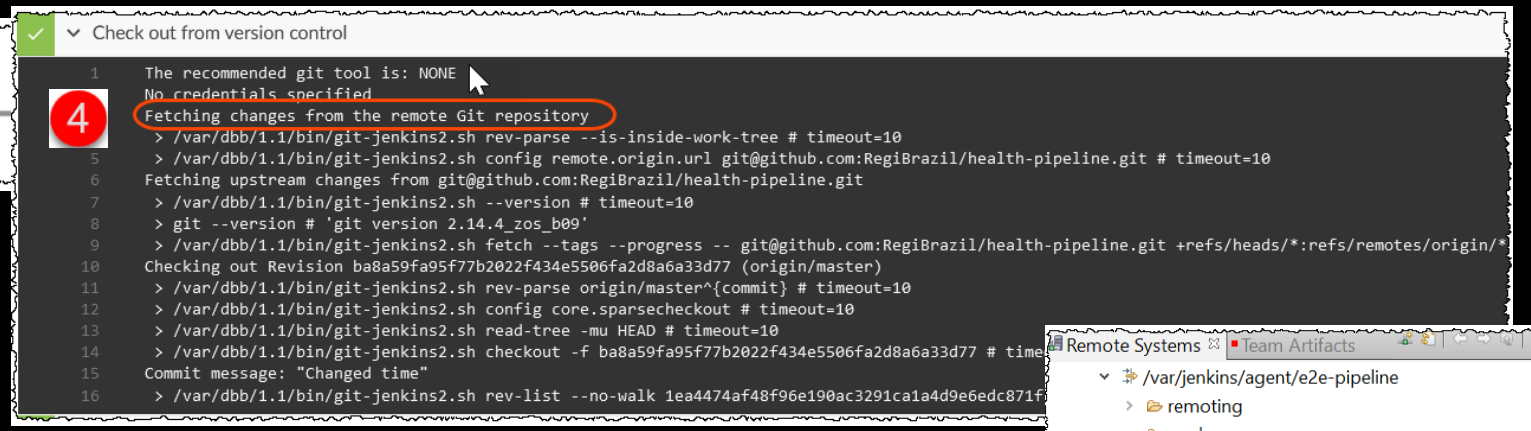
- Free Sample in public Github
  - https://github.com/ibm/dbb
  - https://github.com/IBM/dbb-zappbuild
- Use as starting point for customizing a build environment
- Separates properties (configuration) from scripts
- Contains a simple application, which can also be used for a PoC, if there is no client application available
  - Ex: Mortgage-SA
- Encourage the field as well as clients to contribute to zAppBuild
  - Everyone can contribute because it is in a public repository
  - DBB team keeps track of direction, etc.
  - Everyone can see what currently happens
  - New features are continuously added into the development branch

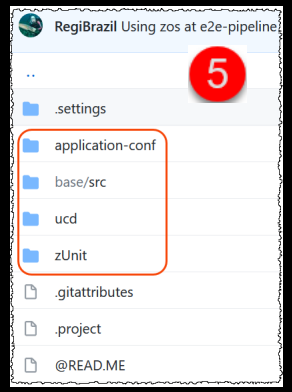
Example: Jenkins Pipeline details using "Jenkinsfile"

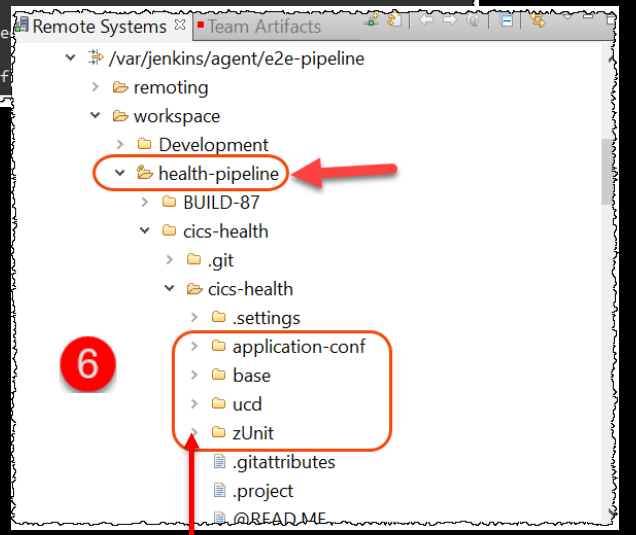




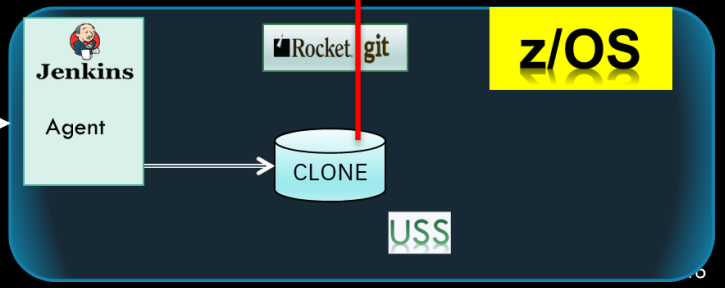




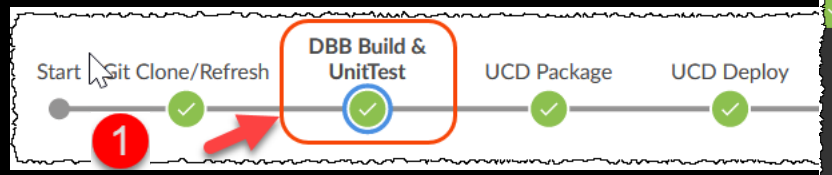




CLONE to Z



Example: Jenkins Pipeline details using "Jenkinsfile"



*Jenkinsfile

```
000151        stage('DBB Build & UnitTest') {
000152            steps {
000153                echo "Perform the Build and Unit
000154                script{
000155                    def zUnitContents = []
000156                    node( zOsAgentLabel ) {
```

2

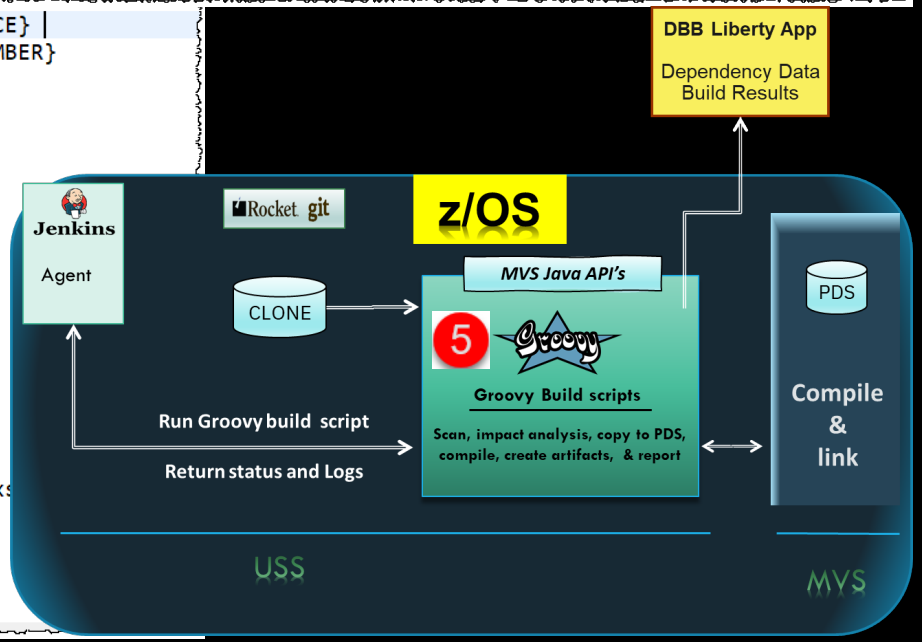
Shell Script

4

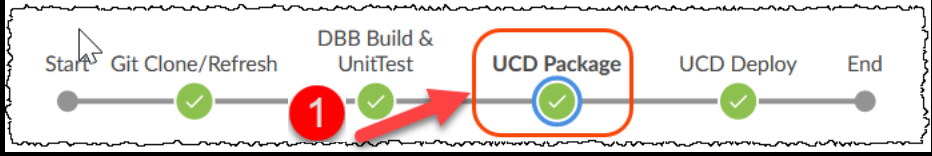
```
+ /var/dbb/1.1/bin/groovyz /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/dbb-zappbuild/build.groovy --logEncoding UTF-8 -w /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline
--application cics-health --sourceDir /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/cics-health --workDir /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87 --hlq
IBMUSER.DBB.HEALTH --url https://clmweb:11043/dbb/ -pw ADMIN -i
Cannot contact e2e-pipeline: java.lang.InterruptedException

** Build start at 20210511.073724.037
** Repository client created for https://clmweb:11043/dbb/
** Build output located at /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87/build.20210511.073724.037
** Build result created for BuildGroup:cics-health-master BuildLabel:build.20210511.073724.037 at https://clmweb:11043/dbb/rest/buildResult/11913
** --impactBuild option selected. Building impacted programs for application cics-health
** Writing build list file to /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87/build.20210511.073724.037/buildList.txt
** Invoking build scripts according to build order: BMS.groovy,Cobol.groovy
** Invoking test scripts according to test order: ZunitConfig.groovy
** Building files mapped to BMS.groovy script
*** Building file cics-health/base/src/BMS/HCMAPS.bms
** Building files mapped to Cobol.groovy script
*** Building file cics-health/base/src/COBOL/HCM1PL01.cbl
*** Building file cics-health/base/src/COBOL/HCT1PL01.cbl
*** Building file cics-health/base/src/COBOL/HCV1PL01.cbl
*** Building file cics-health/base/src/COBOL/HCMAPL01.cbl
*** Building file cics-health/base/src/COBOL/HCAZMENU.cbl
*** Building file cics-health/base/src/COBOL/HCP1PL01.cbl
** Writing build report data to /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87/build.20210511.073724.037/BuildReport.json
** Writing build report to /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87/build.20210511.073724.037/BuildReport.html
** Updating build result BuildGroup:cics-health-master BuildLabel:build.20210511.073724.037 at https://clmweb:11043/dbb/rest/buildResult/11913
** Build ended at Tue May 11 19:40:27 GMT 2021
** Build State : CLEAN
** Total files processed : 7
** Total build time  : 3 minutes, 3.004 seconds

** Build finished
```

```
000169
000170 sh "$dbbHome/bin/groovyz $dbbGroovyzOpts ${WORKSPACE}/dbb-zappbuild/build.groovy --logEncoding UTF-8 -w ${WORKSPACE}
000171           --application cics-health --sourceDir ${WORKSPACE}/cics-health  --workDir ${WORKSPACE}/BUILD-${BUILD_NUMBER}
000172           --hlq ${dbbHlq}.HEALTH --url $dbbUrl -pw ADMIN $dbbBuildType $buildVerbose $dbbBuildExtraOpts"
000173                     def files = findFiles(glob: "**BUILD-${BUILD_NUMBER}/**/buildList.txt")
000174                     // Do not enter into some steps if nothing in the build list
000175                     hasBuildFiles = files.length > 0 && files[0].length > 0
000176
000177                     def zUnitFiles = findFiles(glob: "**BUILD-${BUILD_NUMBER}/**/*.zunit.report.log")
000178                     zUnitFiles.each { zUnit ->
000179     println "Process zUnit: $zUnit.path"
000180                         def zUnitContent = readFile file: zUnit.path
000181                         zUnitContents << zUnitContent
000182                     }
000183                 }
000184                 zUnitContents.each { zUnitContent ->
000185                     writeFile file: '/tmp/zUnit.zunit', text:zUnitContent
000186                     def rc = sh (returnStatus: true, script: '''#!/bin/sh
000187                     curl --silent https://raw.githubusercontent.com/ibm/dbb-pipeline/master/cics-genapp/zUnit/xs
000188                         xsltproc /tmp/AZUZ2J30.xsl /tmp/zUnit.zunit > ${WORKSPACE}/zUnit.xml
000189                     ''')
000190                     junit "zUnit.xml"
000191                 }
```

3

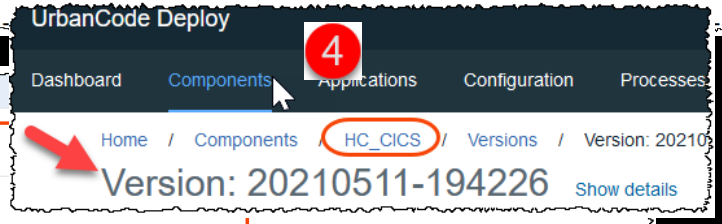


Example: Jenkins Pipeline details using "Jenkinsfile"




```
000210 //*
000211         stage('UCD Package') {
000212             steps {
000213                 echo "Move the binaries created by build to UCD to be deployed"
000214                 script {
000215                     node( zOsAgentLabel ) {
000216                         if ( hasBuildFiles ) {
000217                             def ucdBuztool = env.UCD_BUZTOOL_PATH
000218 // Regi - print UCD Buzztool path
000219             println "*REGI* UCD BUZTOOL_PATH : ${ucdBuztool} "
000220       BUILD_OUTPUT_FOLDER = sh (script: "ls ${WORKSPACE}/BUILD-${BUILD_NUMBER}", returnStdout: true).trim()
000221       dir("${WORKSPACE}/BUILD-${BUILD_NUMBER}/${BUILD_OUTPUT_FOLDER}") {
000222       println "*REGI* Push to UCD Code station"
000223        sh "$dbbHome/bin/groovyz $dbbGroovyzOpts ${WORKSPACE}/dbb/Pipeline/CreateUCDComponentVersion/dbb-ucd-packaging.groovy
000224            --buztool ${ucdBuztool} --component ${ucdComponent} --workDir ${WORKSPACE}/BUILD-${BUILD_NUMBER}/${BUILD_OUTPUT_FOLDER}
000225                             }
000226                         }
000227                     }
000228                 }
000229             }
000230         }
000231
000232 //*
```

2

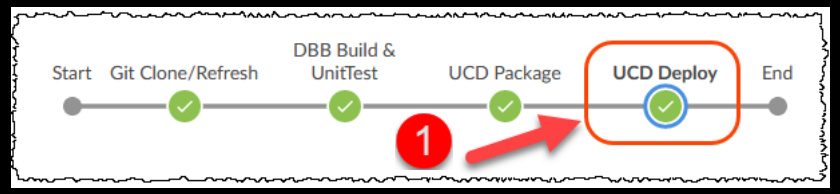


Shell Script

3

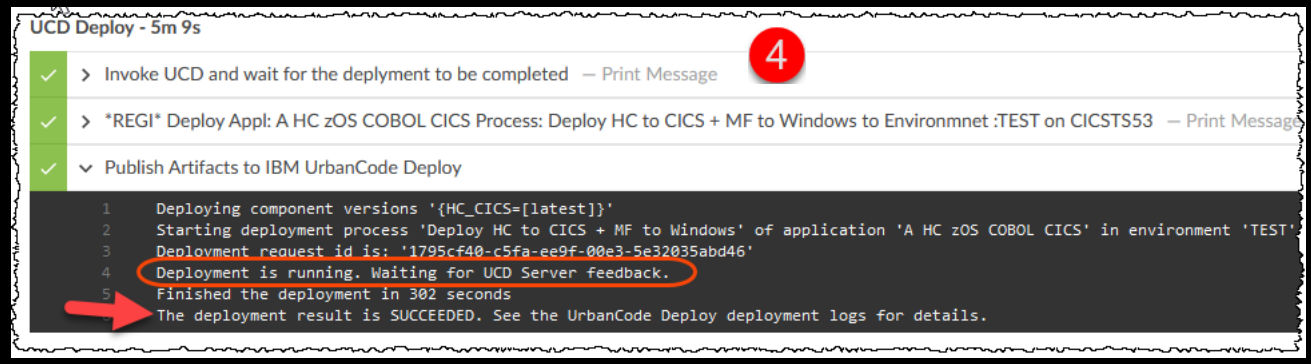
```
+ /var/dbb/1.1/bin/groovyz /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/d
HC_CICS --workDir /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87/b
** Create version start at 20210511.074145.041
** Properties at startup:
   component -> HC_CICS
   startTime -> 20210511.074145.041
   workDir -> /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87/build
   preview -> false
   buztoolPath -> /apps/ucd/v7/bin/buztool.sh
** Read build report data from /var/jenkins/agent/e2e-pipeline/workspace/health-pipeli
Cannot contact e2e-pipeline: java.lang.InterruptedException
** Find deployable outputs in the build report
   ! Buildrecord type TYPE_COPY_TO_PDS is supported with DBB toolkit 1.0.8 and higher.
   IBMUSER.DBB.HEALTH.LOAD(HCMAPS), MAPLOAD
   IBMUSER.DBB.HEALTH.LOAD(HCM1PL01), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCT1PL01), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCV1PL01), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCMAPL01), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCAZMENU), LOAD
   IBMUSER.DBB.HEALTH.LOAD(HCP1PL01), LOAD
** Generate UCD ship list file
** Write ship list file to  /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/
** Following UCD buztool cmd will be invoked
/apps/ucd/v7/bin/buztool.sh createzosversion -c HC_CICS -s /var/jenkins/agent/e2e-pipe
/workspace/health-pipeline/BUILD-87/build.20210511.073724.037/buztool.output
** Create version by running UCD buztool
zOS toolkit config   : /apps/ucd/v7/ (7.0.2.0,20190131-0837)
zOS toolkit binary   : /apps/ucd/v7/ (7.0.2.0,20190131-0837)
zOS toolkit data set : BUZV7 (7.0.2.0,20190131-0837)
Reading parameters:
....Command : createzosversion
....Component : HC_CICS
....Generate version name : 20210511-194226
....Shiplist file : /var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87
....Output File:/var/jenkins/agent/e2e-pipeline/workspace/health-pipeline/BUILD-87/bui
Verifying version
....Repository location : /apps/ucd/v7/var/repository/HC_CICS/20210511-194226
Pre-processing shiplist:
....Shiplist after processing :/apps/ucd/v7/var/repository/HC_CICS/20210511-194226/shi
Packaging data sets:
....Location to store zip : /apps/ucd/v7/var/repository/HC_CICS/20210511-194226
....Zip name : package.zip
....IBMUSER.DBB.HEALTH.LOAD.bin
....Elapsed time for data set package or deploy operation : 5.544010
```

Example: Jenkins Pipeline details using "Jenkinsfile"



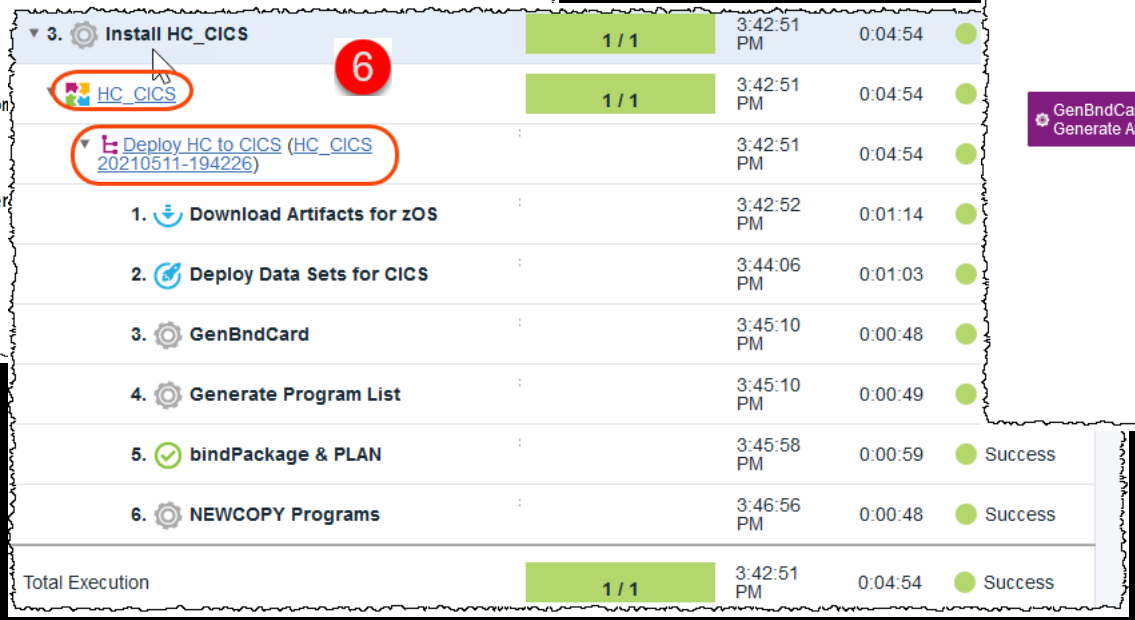

```
000037
000038 // UCD
000039 def ucdApplication = 'A HC zOS COBOL CICS'
000040 def ucdProcess = 'Deploy HC to CICS + MF to Windows'
000041 def ucdComponent = 'HC_CICS'
000042 def ucdEnv = 'TEST'
000043 def ucdSite = 'Urbancode server'
```

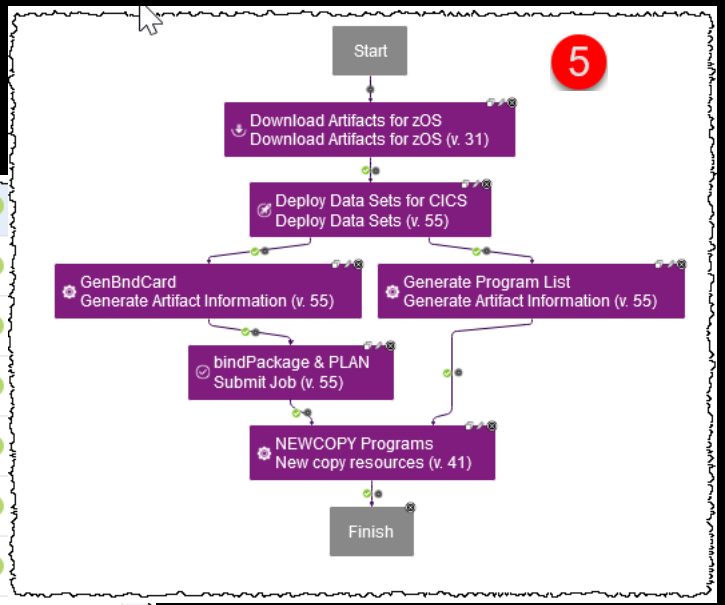
2




```
000231 //* ------------------------------------------------------------------------------------------
000232         stage('UCD Deploy') {
000233             steps {
000234                 echo "Invoke UCD and wait for the deplyment to be completed"
000235     println "*REGI* Deploy Appl: ${ucdApplication} Process: ${ucdProcess}  to Environmnet :${ucdEnv}  on CICSTS53"
000236                 script{
000237                     if ( hasBuildFiles ) {
000238                         script{
000239                             step(
000240                                 [$class: 'UCDeployPublisher',
000241                                   deploy: [
000242                                       deployApp: ucdApplication,
000243                                       deployDesc: 'Requested from
000244                                       deployEnv: ucdEnv,
000245                                       deployOnlyChanged: false,
000246                                       deployProc: ucdProcess,
000247                                       deployVersions: ucdComponen
000248                                   siteName: ucdSite])
000249                         }
000250                     }
000251                 }
000252             }
000253         }
```

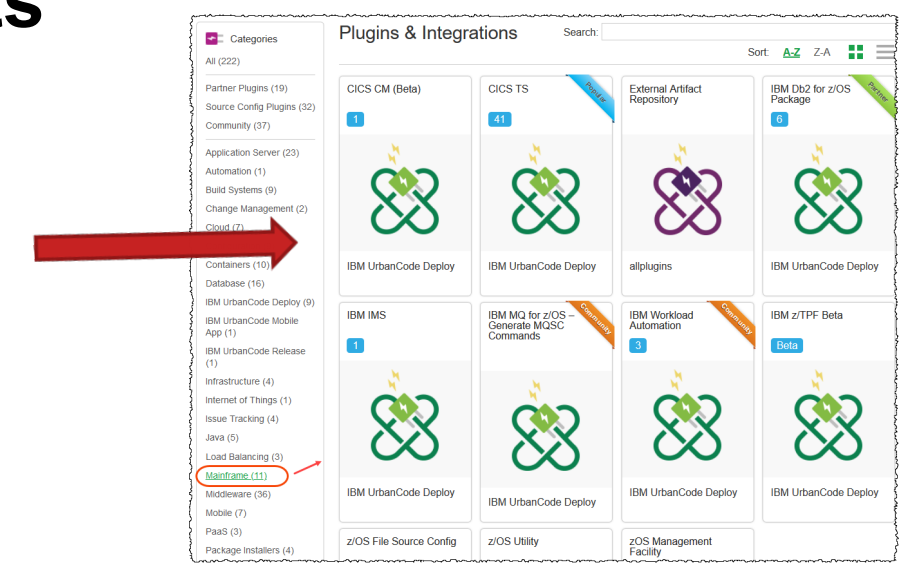
3

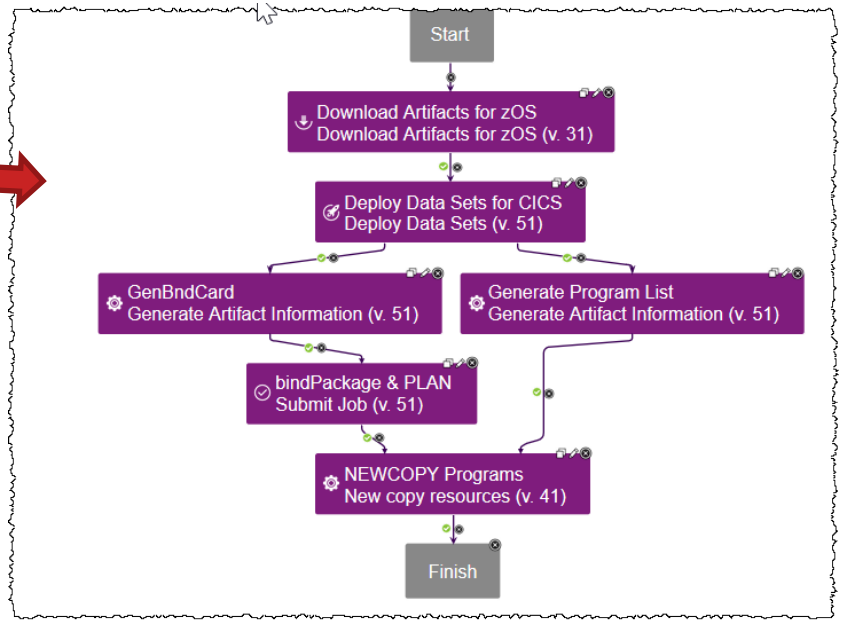




IBM UrbanCode Deploy on Z versus Scripts

- Z/OS agents that allow security implementations at many levels (RACF integration)
- It is a deploy tool with many z/OS plugins like CICS Newcopy, Submit JCL, FTP, RACF, MQ, etc… No need to write scripts.
- Allow rollbacks with minimum effort (automatically keep backups) of deployed code
- Incremental deploy (deploy only a changed loadlib)
- Makes deployment process easy to understand
- Keep repository of "binaries" and integrates with *Nexus/Artifactory*
- Allow many levels of authorizations and deploy configurations
- Auditing capabilities
- Include external activities in a deployment process, like get approvals to deploy to prod





Interesting papers

1. Advanced Build and Migration recipes with DBB and zAppBuild

   https://www.ibm.com/support/pages/node/6427617

2. Build a pipeline with Jenkins, Dependency Based Build, and UrbanCode Deploy

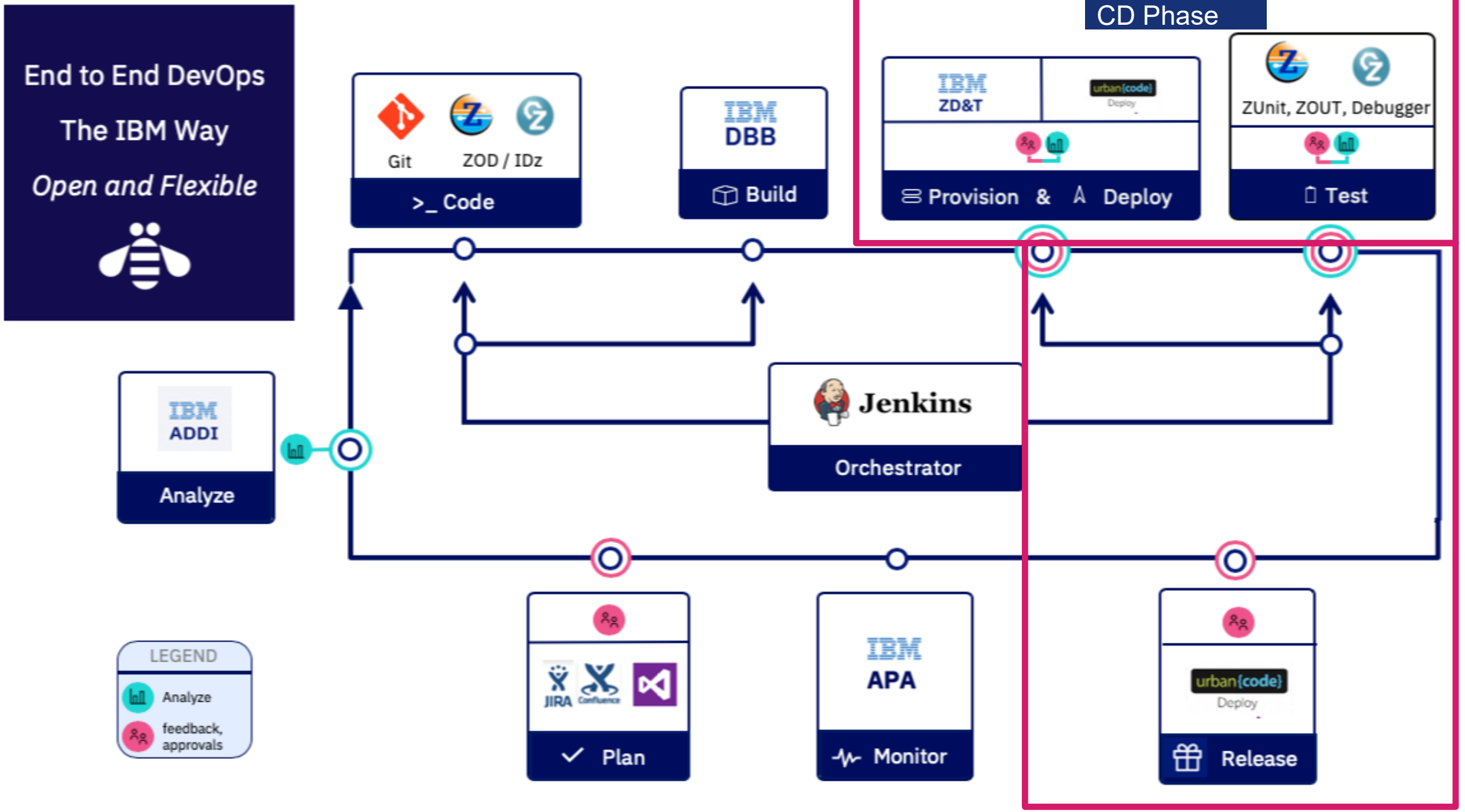
   https://developer.ibm.com/components/ibmz/tutorials/build-a-pipeline-with-jenkins-dependency-based-build-and-urbancode-deploy/

3. Build a pipeline with GitLab CI, IBM Dependency Based Build, and IBM UrbanCode Deploy

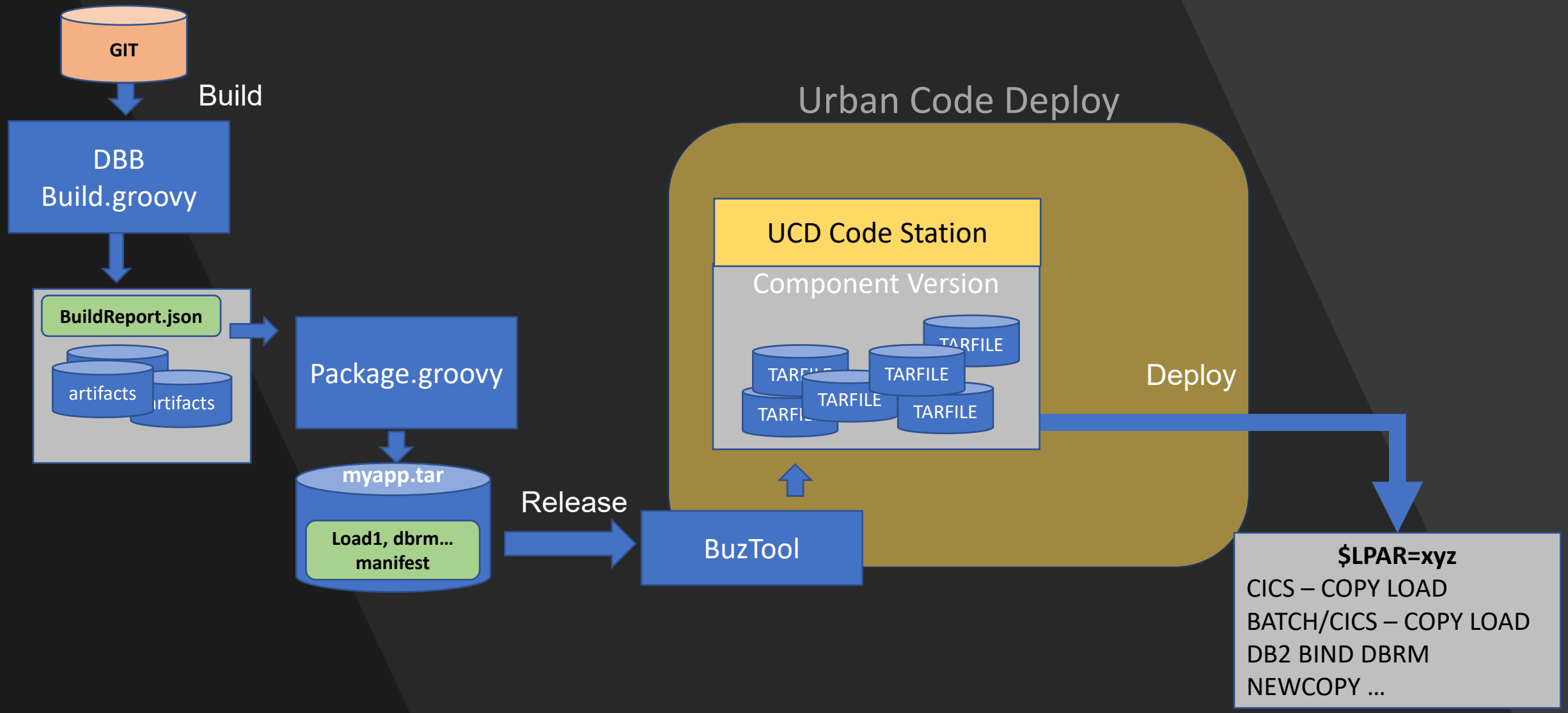
   https://developer.ibm.com/tutorials/build-a-pipeline-with-gitlab-ci-dbb-and-ucd/

Thank You

BACKUP

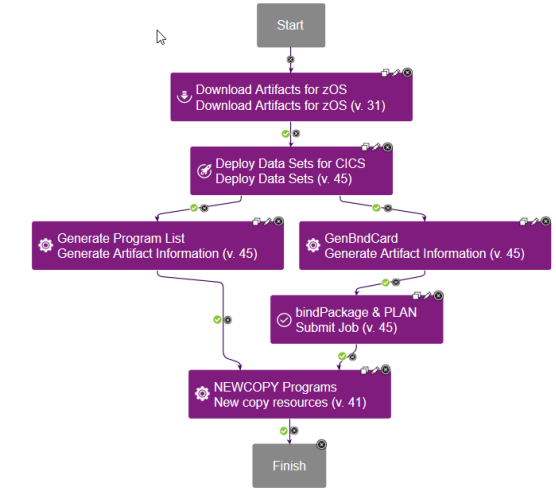

End to End DevOps
The IBM Way
Open and Flexible
CD Phase
Git
ZOD / IDz
>_ Code
IBM DBB
Build
IBM ZD&T
urban{code} Deploy
Provision & Deploy
ZUnit, ZOUT, Debugger
Test
IBM ADDI
Analyze
Jenkins
Orchestrator
JIRA
Confluence
Plan
IBM APA
Monitor
urban{code} Deploy
Release
LEGEND
Analyze
feedback, approvals

Sample CICS - Build, Package and Deploy Flow



URBANCODE DEPLOY FOR DEPLOYMENT AUTOMATION

DEPLOYING TO MAINFRAME



Develop

SCM

Build

IDE
or
TSO/ISPF

CA Endevor

Serena Changeman

git

RTC

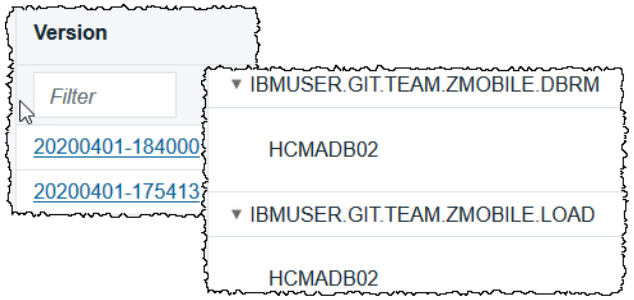
EE

Jenkins

IBM UrbanCode Deploy

From Source Code,
Using JCL:
Compile, Link and **Bind**

LOADLIB + DBRM



Dev

UCD Agent

z/OS LPAR

TEST

UCD Agent

z/OS LPAR

PROD

UCD Agent

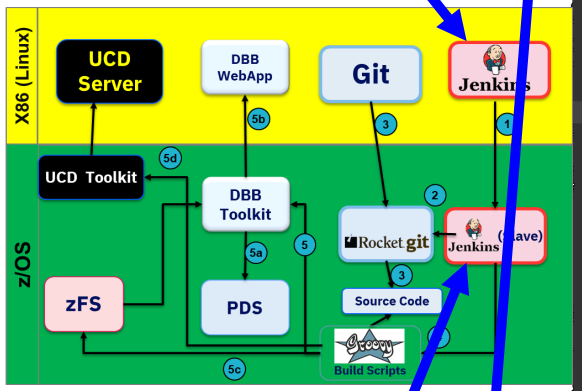
z/OS LPAR

Mainframe

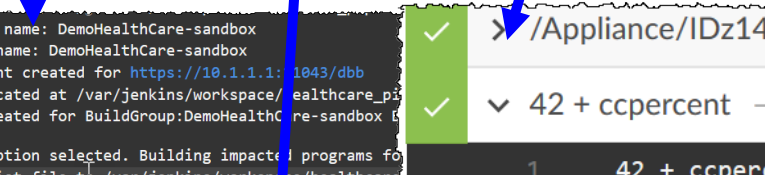
Demo High Level Diagram using Jenkins-Git-DBB-UCD

healthcare_pipeline3_DemoHealthCare_Wazi < 32

Start | Init | Git Clone/Refresh | DBB Build | Unit Tests | Code coverage gating | Wazi VTP - Integration test | Push to UCD Code station | Deploy to CICS using UCD | End



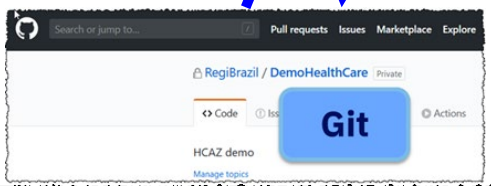

```
** DBB build group name: DemoHealthCare-sandbox
** DBB collection name: DemoHealthCare-sandbox
** Repository client created for https://10.1.1.1:1043/dbb
** Build output located at /var/jenkins/workspace/healthcare_pi
** Build result created for BuildGroup:DemoHealthC re-sandbox
/buildResult/7783
** --impactBuild option selected. Building impacted programs fo
** Writing build list file to /var/jenkins/workspace/healthcare
** Invoking build scripts according to build orde : BMS.groovy,
** Building files mapped to BMS.groovy script
** Building files mapped to Cobol.groovy script
*** Building file DemoHealthCare/cobol/vsam_db2/THCIPDB0.cbl
** Building files mapped to LinkEdit.groovy script
** Writing build report data to /var/jenkins/workspace/healthcare_pipeline2/BUILD-24/BuildReport.json
** Writing build report to /var/jenkins/workspace/healthcare_pipeline2/BUILD-24/BuildReport.html
** Build ended at Fri Jul 10 20:51:38 GMT 2020
** Build State : CLEAN
** Total files processed : 1
** Total build time  : 4 minutes, 33.933 seconds

** Build finished
```




```
PackageManifest file post-processing completed.
Create version and store package:
....Version artifacts stored to UCD server CodeStation
....Version:20200828-150944 created
Elapsed time 68.0 seconds.
```

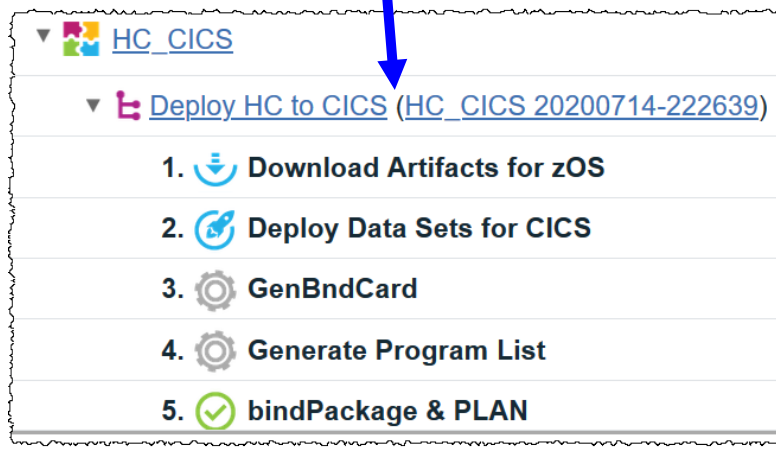
https://www.ibm.com/support/pages/node/6427617

```
BZUP001I EXECUTING RUNNER PROGRAM=BZURUN AT 2020/08/10 11:33:47
POWERED BY IBM WAZI VIRTUAL TEST PLATFORM - V1R1M0
COPYRIGHT(C) 2020, IBM CORP. ALL RIGHTS RESERVED.
********************************************************************************
BZUP002I FINISHED EXECUTION RC=00
********************************************************************************
```




```
Checking out Revision e375bc120265eafdc47511e9d65b17780124c322 (origin/sandbox)
 > git rev-parse origin/sandbox^{commit} # timeout=10
 > git config core.sparsecheckout # timeout=10
 > git config core.sparsecheckout true # timeout=10
 > git read-tree -mu HEAD # timeout=10
 > git checkout -f e375bc120265eafdc47511e9d65b17780124c322 # timeout=10
Commit message: "Fixed Bug"
 > git rev-list --no-walk 40455f54e22d7d8b05d951659de3ddeab6d67605 # timeout=10
```

```
The JCL Job RUNZUNIT(JOB00179) completed successfully
********** Module [THCIPDB0] **********
Name:      THCIPDB0
Status:    PASS
Test cases: 2 (2 passed, null failed, 0 errors)
Details:
    TEST2
    TEST3
********** Module [THCIPDB0] **********
** Build finished
```



* UCD = UrbanCode Deploy