z/TPF EE V1.1
z/TPFDF V1.1
TPF Toolkit for WebSphere® Studio V3
TPF Operations Server V1.2

IBM Software Group

TPF Users Group Spring 2006

Migrating applications from TPF 4.1 to z/TPF using IBM TPF Toolkit

Name: Mary Huang
Venue: Education Session

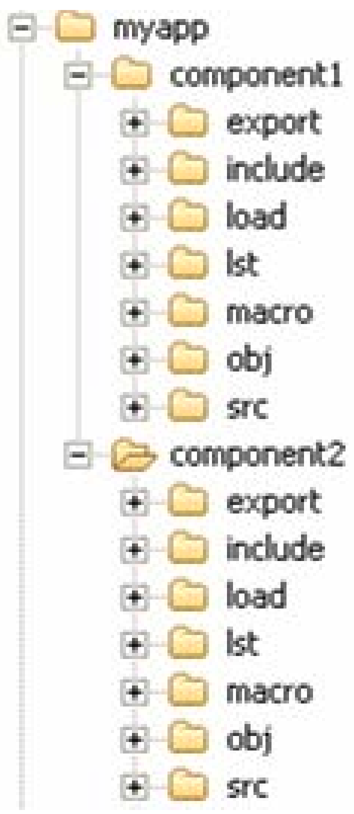
AIM Enterprise Platform Software
IBM z/Transaction Processing Facility Enterprise Edition 1.1.0

Application Migration

1. Move your application source to HFS
2. Convert to MakeTPF based builds
3. Convert to Single Source
4. Compile code for TPF 4.1
5. Compile code for z/TPF
6. Build and test code on TPF 4.1
7. Build and test code on z/TPF

Moving source code to HFS

- Move source code to USS or Linux
  - Consider moving source directly to Linux
  - Mount the Linux location on USS to access source files on USS.
- Define HFS directory
  - Things to consider: build environment, SCM, project structure
  - one directory for each application
  - multiple application directories, divided by function
  - use sub-directories to set up logical groupings
  - consider defining a directory structure that can be used for both TPF 4.1 and z/TPF

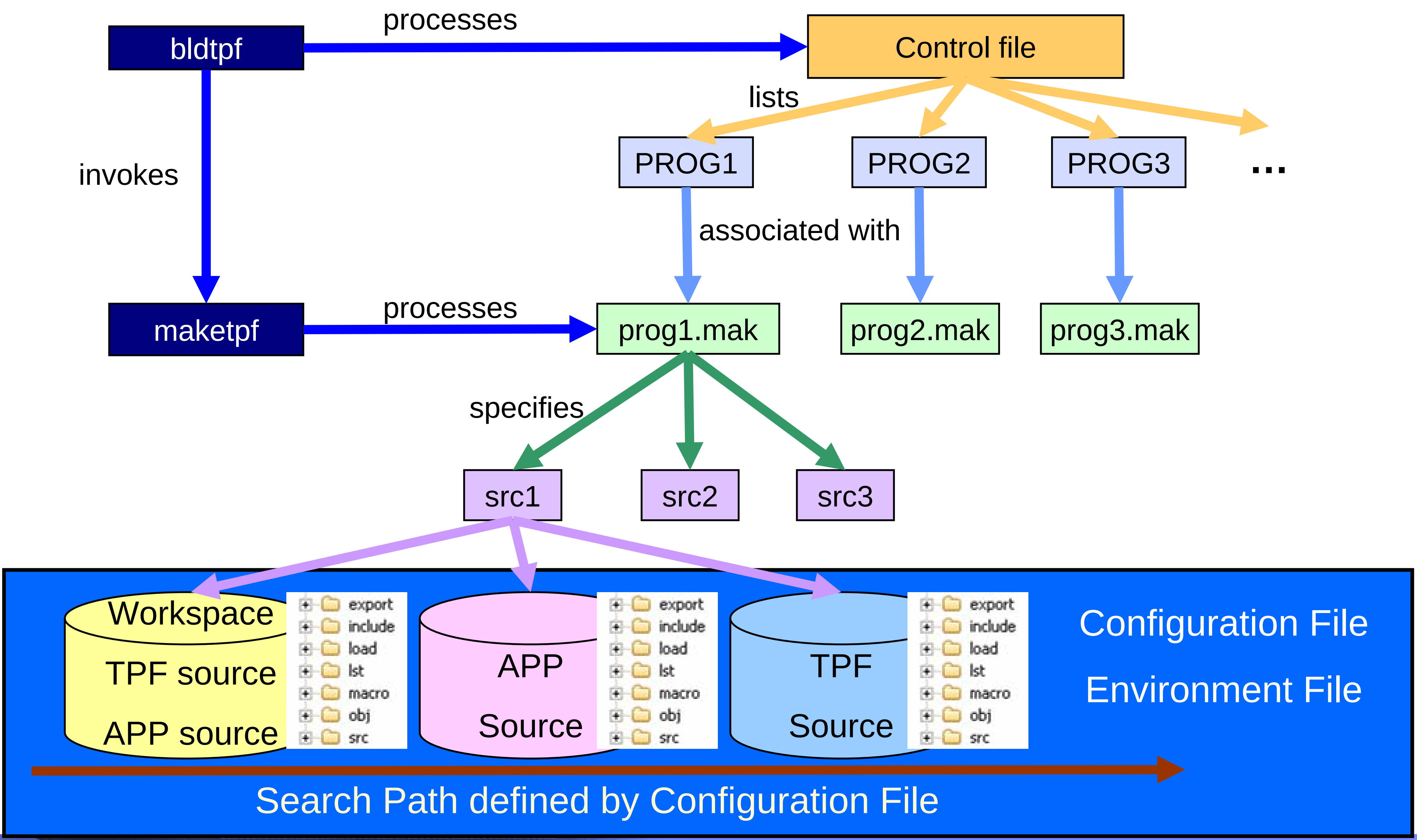


```
myapp
  component1
    export
    include
    load
    lst
    macro
    obj
    src
  component2
    export
    include
    load
    lst
    macro
    obj
    src
```

Converting to MakeTPF based build

- Why MakeTPF?
  - Recommended build tool set for z/TPF
  - Same tools and makefiles for production and development builds
  - Available on Linux and USS for both online and offline program builds
  - TPF Toolkit supports MakeTPF based builds for TPF 4.1 and z/TPF
  - Minimal makefile knowledge required

Overview of MakeTPF



Converting to MakeTPF based builds

- Understand MakeTPF
  - TPF Information Center
    - Program Management book
    - SYSGEN book
  - manpages
  - MakeTPF education sessions at previous TPFUG
  - download MakeTPF Build Solution for TPF 4.1 sample

  http://www.ibm.com/software/htp/tpf/download/maketpf.htm

Converting to MakeTPF based builds

- Setting up the environment for using MakeTPF
  - Create environment files
  - Create Makefiles
  - Create Configuration files
  - Create Control files

Environment file

- Map HFS directory structure to environment files
- one environment file per unique HFS directory structure
- consider one environment file per application directory
  - one environment file per component if application source tree has subdirectory for each component
  - describes the directory structure per application
  - enables directory structure update of an application without affecting others
- can re-use environment files in z/TPF if same HFS directory structure is maintained
- use maketpf.env_myappls as sample

Makefiles

- each target program needs a makefile
  - unique makefile per program
  - generic makefile for single segment BAL programs if program name and segment name are same
- a makefile specifies the program name, program type, build options override for program or segments, etc
- create new or convert from existing build scripts
  - Wizard in TPF Toolkit to generate makefiles
  - Wizard in TPF Toolkit to convert build scripts to makefiles

Configuration Files

- defines build space, root source directories, provides build options override
- consider one configuration file per target system
  - maps to TPF Make Configuration option set in TPF Toolkit Target Environment

Control Files

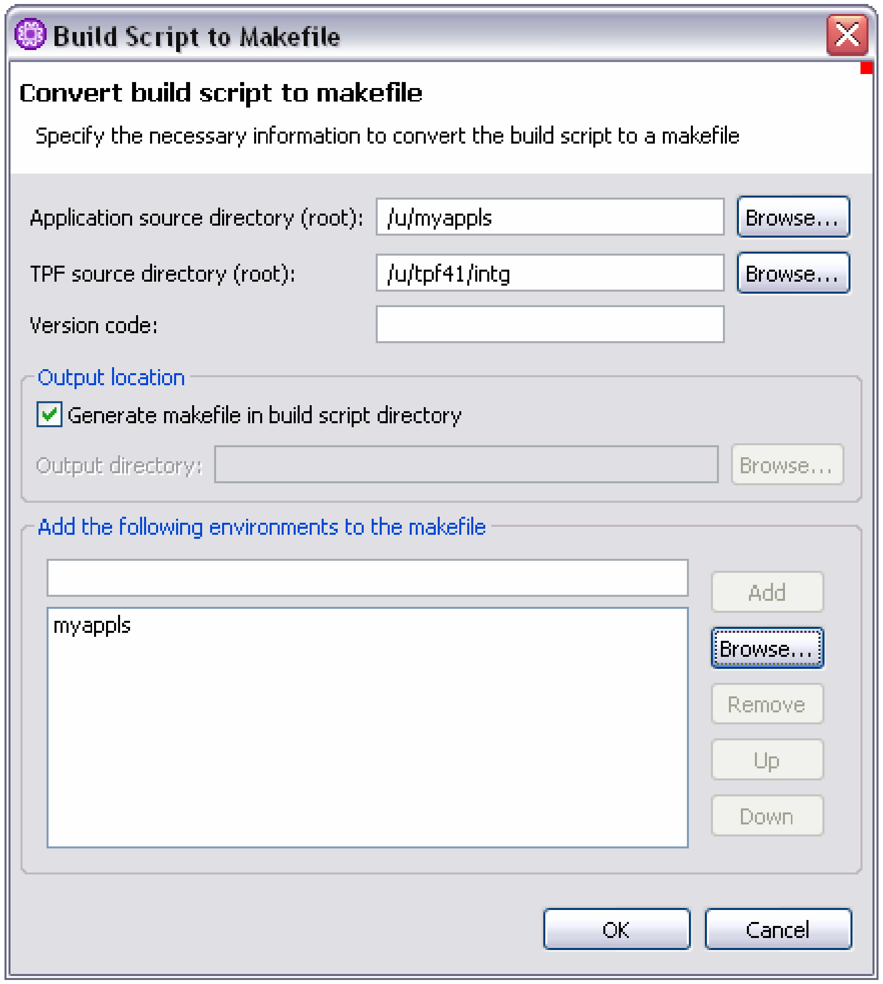
- defines list of programs to build, build order, offline/online, etc
- input for creating loader input files
- can include other control files (one level)
- consider one control file per application
- use one master control file for ALL applications (usr.cntl)
  - include other application specific control files
- `convert_usrtpf2cntl.sh` tool provides high level conversion of usrtpf.cpy file to a control file

Convert build script to makefiles

- Convert build scripts to makefiles (maketpf.bsc.convert)
  - tool to assist converting DLM, DLL and LLM build scripts to MakeTPF makefiles
  - run tool from command line as well as from TPF Toolkit

```
maketpf.bsc.convert bsc [-i appl_roothfs] [-e env_list] [-t tpf_roothfs] [-o dir]
[-v version | -v2] [-p prolog_file]

   bsc is the name of the build script to process
    -i  defines the source hfs to use to locate application source files.
    -e  defines a maketpf_env name to be included in the makefile.
    -v  defines the version code for the application source files
        to be used in addition to the TPF version codes(40, 41, R0, H0, M0)
        when searching.
    -v2 defines the last two characters of any application DSD or object file
        name in the build script are to be treated as a version code.
    -t  defines the source hfs to use to locate TPF source files.
        By default /u/tpf41/intg is used
    -o  defines the output directory to write the makefile.
        By default $PWD is used.
    -p  defines the name of the file containing the prolog to add to the
        beginning of the makefile.
```

Build script to makefile wizard



Convert build script to makefile

Specify the necessary information to convert the build script to a makefile

Application source directory (root): /u/myappls

TPF source directory (root): /u/tpf41/intg

Version code:

Output location

[✓] Generate makefile in build script directory

Output directory:

Add the following environments to the makefile

myappls

build script to makefile

Build script file:

```
DLM QPME41  # Include startup code for DLM (DLL application)

@IMPORTDS QPN841  # Include a definition side-deck

#Object File  Function                           Source Language
#-----------  --------                           ---------------
QPME41        # main()                           C
```

maketpf.bsc.convert

Resulting makefile:

```
APP := QPME
APP_TYPE := DLM

LIBS := QPN8

maketpf_env := myappls
maketpf_env += base_rt
maketpf_env += system

C_SRC := qpme.c

include maketpf.rules
```

build script to makefile

```
DLM QPME41  # Include startup code for DLM (DLL application)

@IMPORTDS QPN841  # Include a definition side-deck

#Object File  Function                           Source Language
#-----------  --------                           ---------------
QPME41        # main()                           C
```

Program name and type (minus the version)

```
APP := QPME
APP_TYPE := DLM

LIBS := QPN8

maketpf_env := myappls
maketpf_env += base_rt
maketpf_env += system

C_SRC := qpme.c

include maketpf.rules
```

build script to makefile

```
DLM QPME41  # Include startup code for DLM (DLL application)

@IMPORTDS QPN841  # Include a definition side-deck

#Object File  Function                          Source Language
#-----------  --------                          ---------------
QPME41        # main()                          C
```

Definition side deck converted to LIBS (minus the version)

```
APP := QPME
APP_TYPE := DLM

LIBS := QPN8

maketpf_env := myappls
maketpf_env += base_rt
maketpf_env += system

C_SRC := qpme.c

include maketpf.rules
```

build script to makefile

```
DLM QPME41  # Include startup code for DLM (DLL application)

@IMPORTDS QPN841  # Include a definition side-deck

#Object File  Function                           Source Language
#-----------  --------                           ---------------
QPME41        # main()                           C
```

Object file converted to C source found in application root directory (minus the version)

```
APP := QPME
APP_TYPE := DLM

LIBS := QPN8

maketpf_env := myappls
maketpf_env += base_rt
maketpf_env += system

C_SRC := qpme.c

include maketpf.rules
```

build script to makefile

```
DLM QPME41  # Include startup code for DLM (DLL application)

@IMPORTDS QPN841  # Include a definition side-deck

#Object File  Function                           Source Language
#-----------  --------                           ---------------
QPME41        # main()                           C
```

```
APP := QPME
APP_TYPE := DLM

LIBS := QPN8

maketpf_env := myappls
maketpf_env += base_rt
maketpf_env += system

C_SRC := qpme.c

include maketpf.rules
```

Application environments

build script to makefile

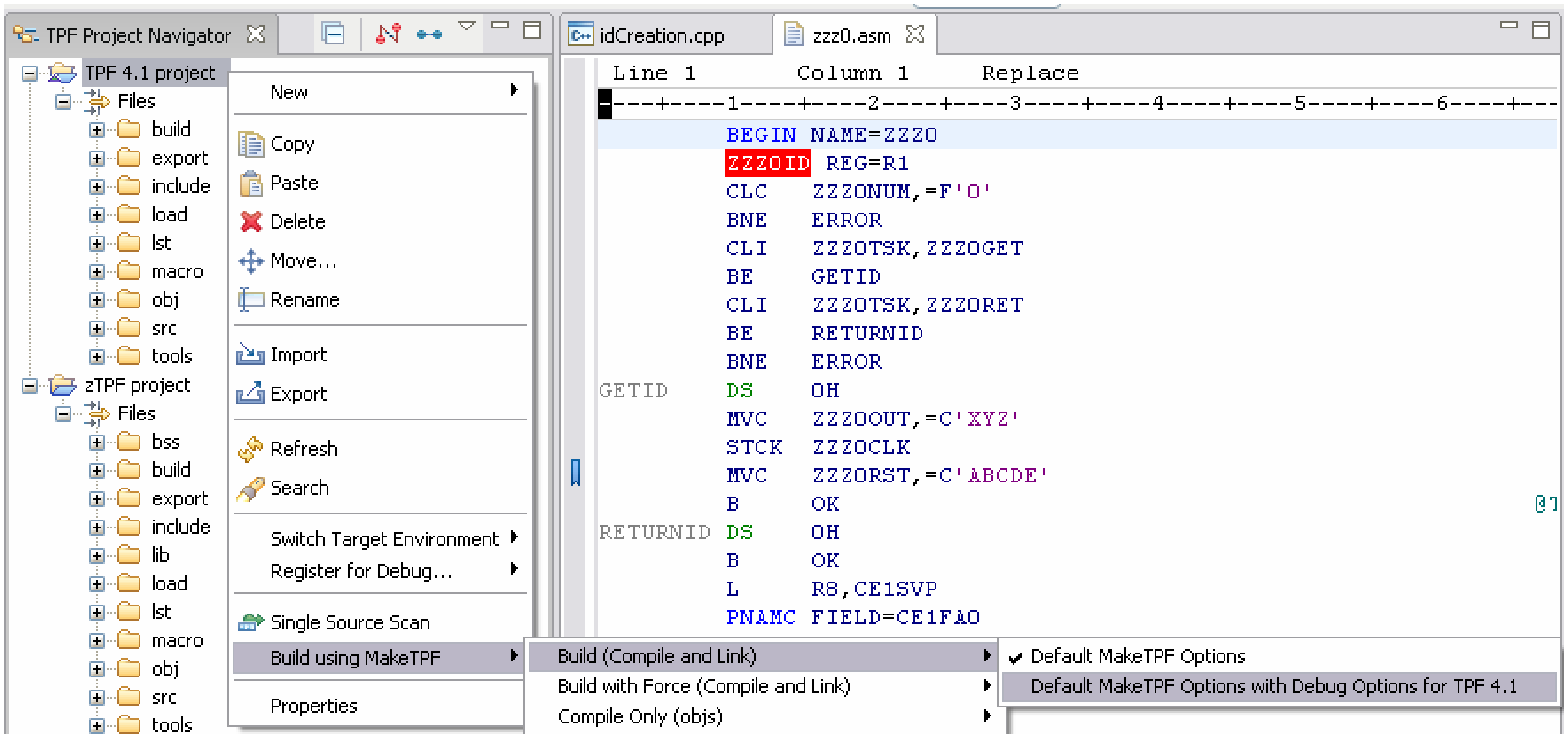
```
DLM QPME41  # Include startup code for DLM (DLL application)

@IMPORTDS QPN841  # Include a definition side-deck

#Object File  Function                   Source Language
#-----------  --------                   ---------------
QPME41        # main()                   C
```

```
APP := QPME
APP_TYPE := DLM

LIBS := QPN8

maketpf_env := myappls
maketpf_env += base_rt
maketpf_env += system

C_SRC := qpme.c

include maketpf.rules
```

Rules included by default

TPF Toolkit supports MakeTPF

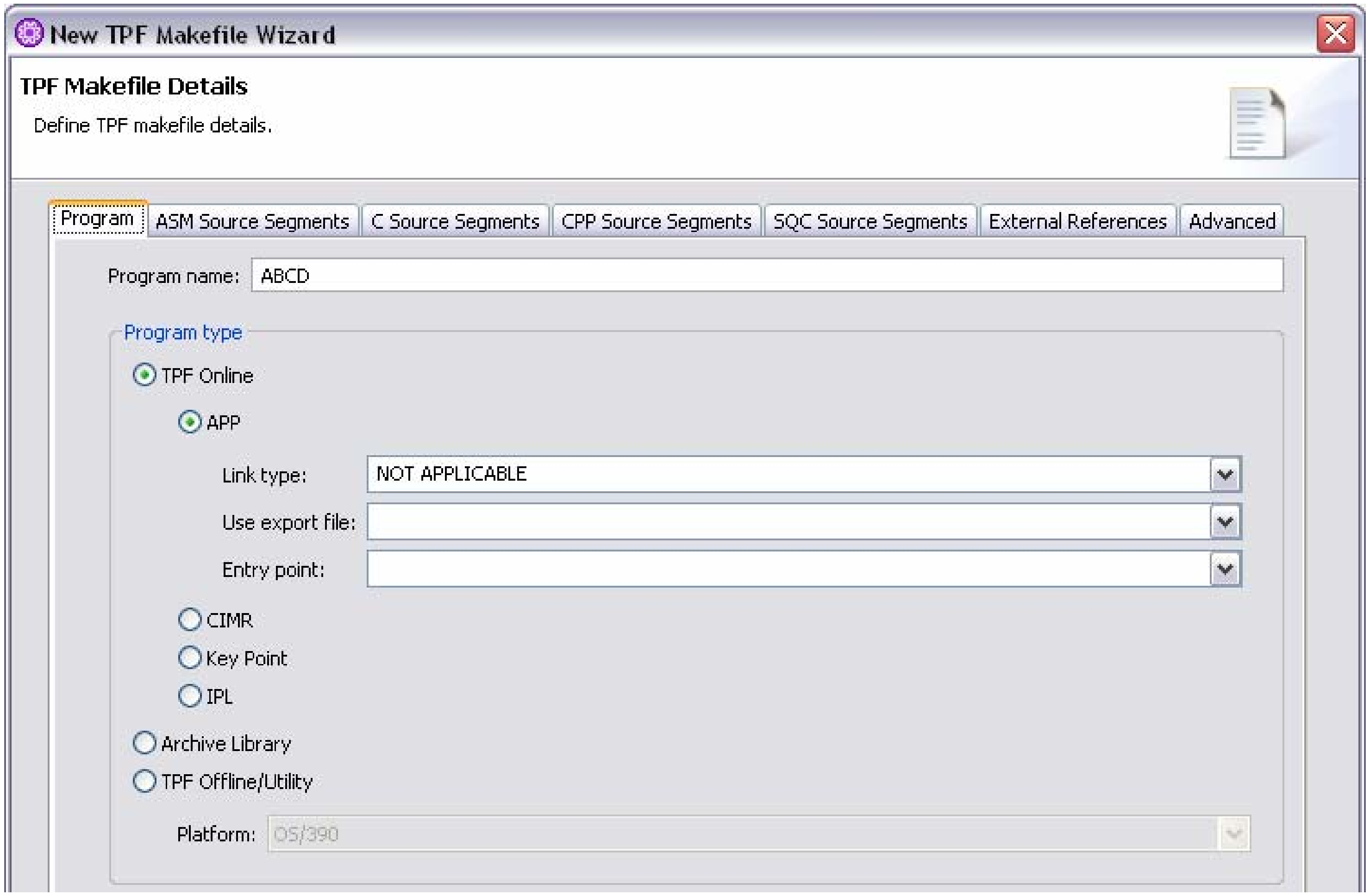
- TPF Toolkit supports maketpf based builds for TPF 4.1 and z/TPF applications



```
 Line 1        Column 1     Replace
----+----1----+----2----+----3----+----4----+----5----+----6----+---
                  BEGIN NAME=ZZZ0
                  ZZZOID REG=R1
                  CLC    ZZZONUM,=F'0'
                  BNE    ERROR
                  CLI    ZZZOTSK,ZZZOGET
                  BE     GETID
                  CLI    ZZZOTSK,ZZZORET
                  BE     RETURNID
                  BNE    ERROR
GETID             DS     0H
                  MVC    ZZZOOUT,=C'XYZ'
                  STCK   ZZZOCLK
                  MVC    ZZZORST,=C'ABCDE'
                  B      OK
RETURNID          DS     0H
                  B      OK
                  L      R8,CE1SVP
                  PNAMC  FIELD=CE1FA0
```

TPF Toolkit supports MakeTPF

- Wizard to generate makefiles



New TPF Makefile Wizard

TPF Makefile Details

Define TPF makefile details.

Program name: ABCD

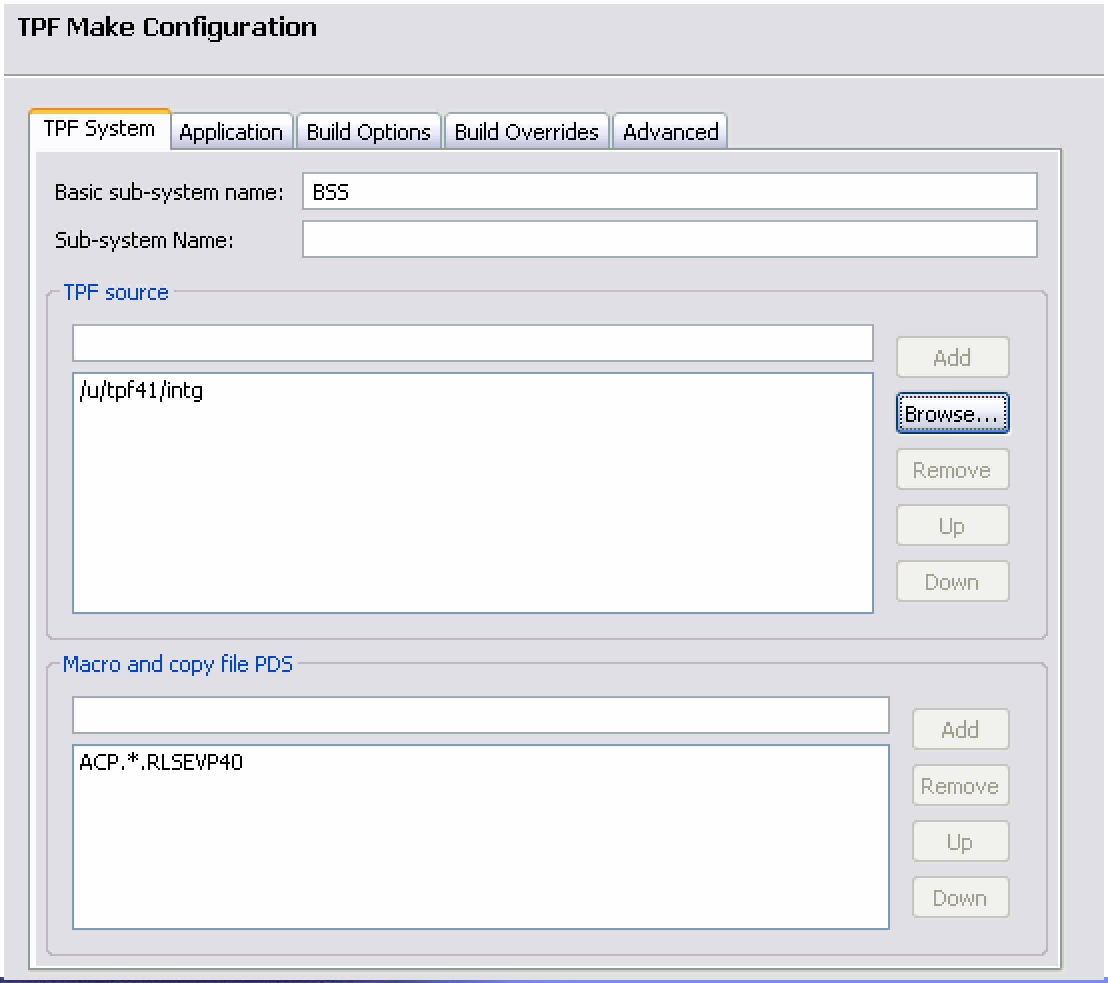
Program type

- [✓ TPF Online]
  - [✓ APP]
    - Link type: NOT APPLICABLE
    - Use export file:
    - Entry point:
  - CIMR
  - Key Point
  - IPL
- Archive Library
- TPF Offline/Utility

Platform: OS/390

TPF Toolkit supports MakeTPF

- Wizard to generate configuration files



TPF Make Configuration

Basic sub-system name: BSS

Sub-system Name:

TPF source

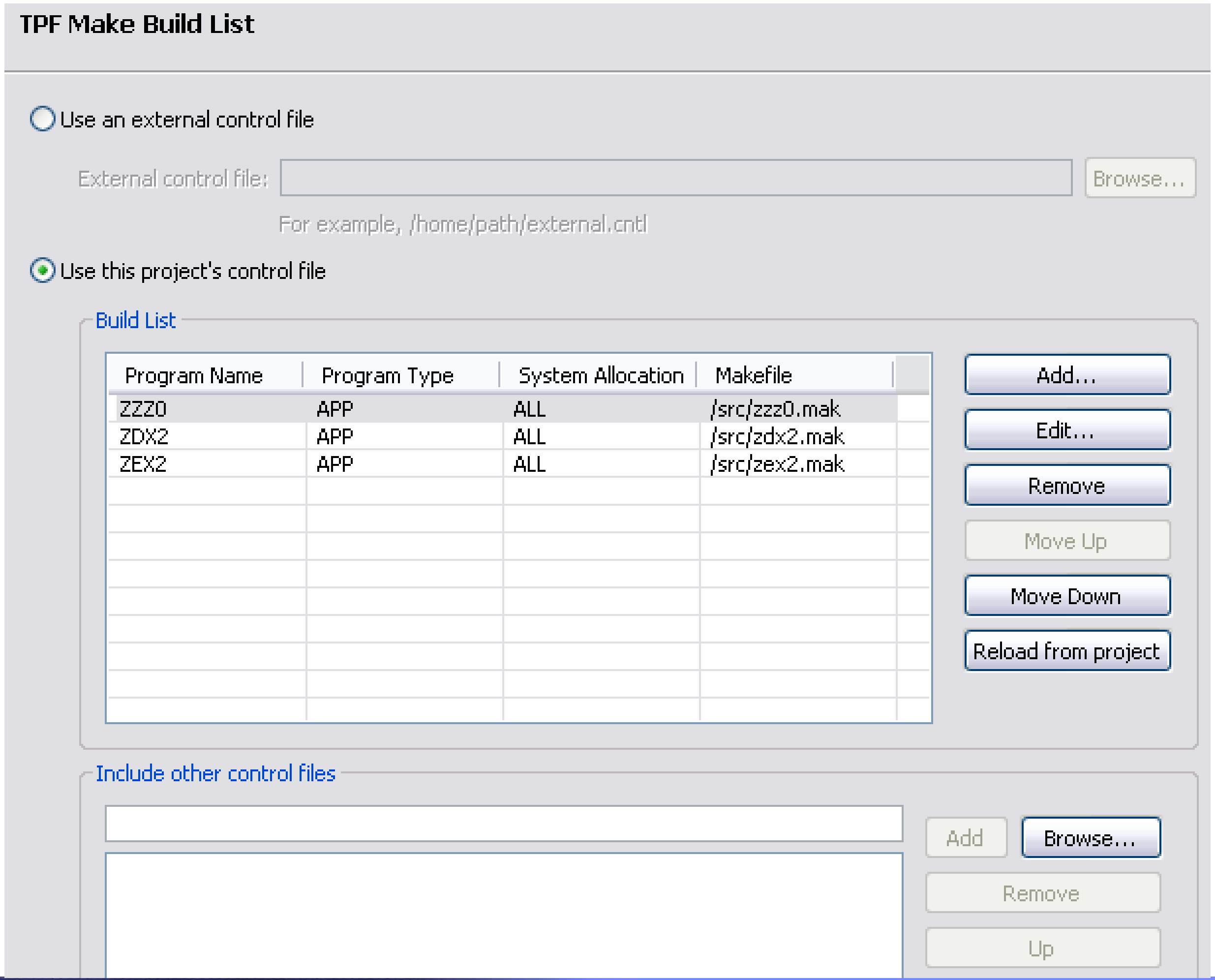
/u/tpf41/intg

Macro and copy file PDS

ACP.*.RLSEVP40

TPF Toolkit supports MakeTPF

- wizard to generate control files



TPF Make Build List

( ) Use an external control file

External control file: [ ] Browse...

For example, /home/path/external.cntl

(●) Use this project's control file

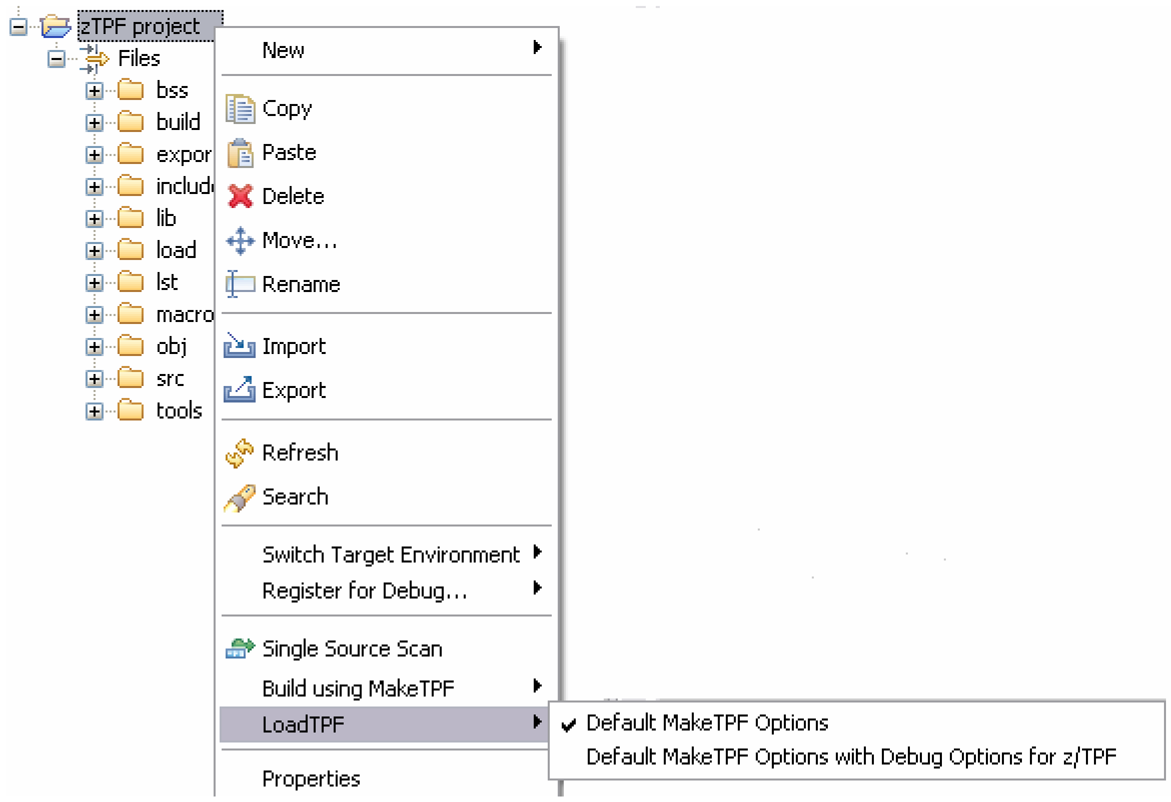
Build List

Program Name	Program Type	System Allocation	Makefile
ZZZ0	APP	ALL	/src/zzz0.mak
ZDX2	APP	ALL	/src/zdx2.mak
ZEX2	APP	ALL	/src/zex2.mak

Include other control files

TPF Toolkit supports MakeTPF

- Load to z/TPF using loadtpf
  - supports OLD and TLD (z/TPF PUT 02)



Trademarks