

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

© IBM Corporation 2006

Any references to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

# 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

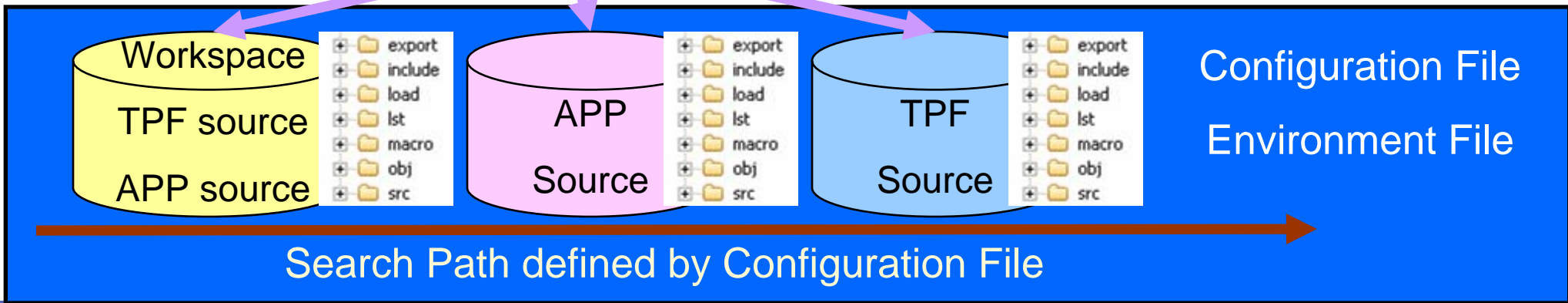
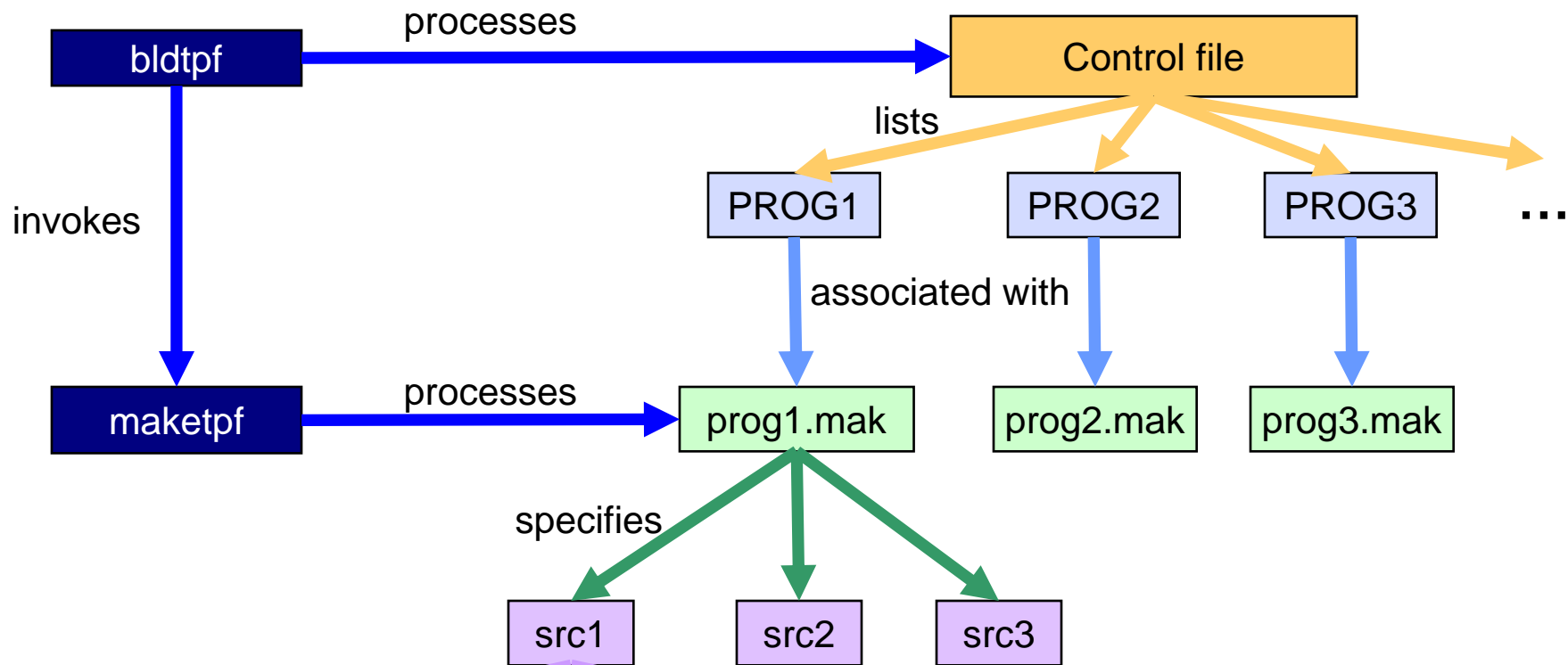
- Move source code to USS or Linux
  - Consider moving source directly to Linux
  - Mount the Linux location on USS to access source files on USS.
- Define HFS directory
  - Things to consider: build environment, SCM, project structure
  - one directory for each application
  - multiple application directories, 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



# 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/maketspf.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

The screenshot shows a dialog box titled "Build Script to Makefile" with a close button (X) in the top right corner. The main heading is "Convert build script to makefile" with the instruction "Specify the necessary information to convert the build script to a makefile".

The dialog contains the following fields and controls:

- Application source directory (root):** A text box containing "/u/myappls" and a "Browse..." button.
- TPF source directory (root):** A text box containing "/u/tpf41/intg" and a "Browse..." button.
- Version code:** An empty text box.
- Output location:** A section with a checked checkbox "Generate makefile in build script directory" and an "Output directory:" text box with a "Browse..." button.
- Add the following environments to the makefile:** A list box containing "myappls". To the right of the list are buttons for "Add", "Browse...", "Remove", "Up", and "Down".

At the bottom of the dialog are "OK" and "Cancel" buttons.

# 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 := myapps
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
```

## Application environments

```
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
```

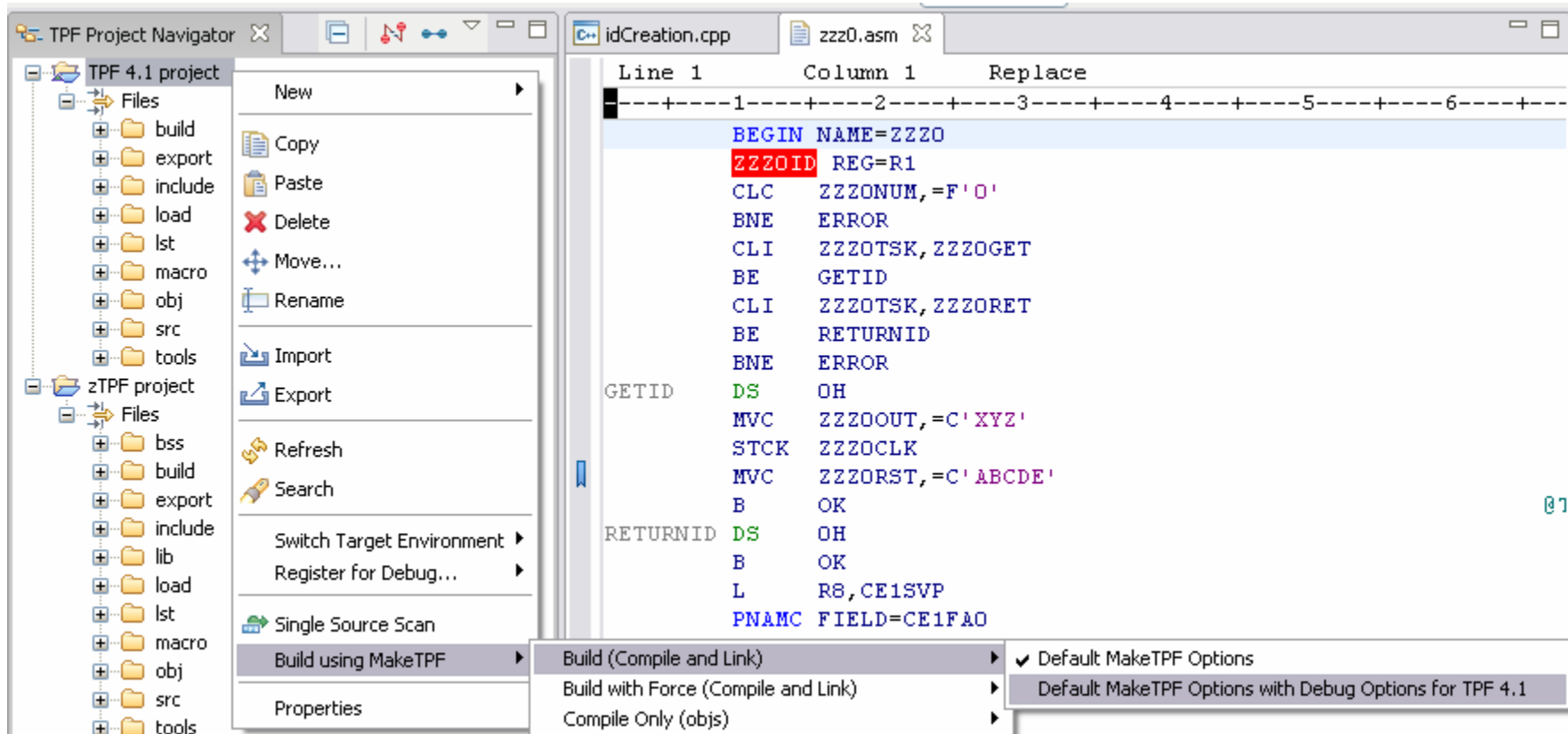
```
maketpf_env := myappls
maketpf_env += base_rt
maketpf_env += system
```

```
C_SRC := qpme.c Rules included by default
```

```
include maketpf.rules
```

# TPF Toolkit supports MakeTPF

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



# TPF Toolkit supports MakeTPF

- Wizard to generate makefiles

**New TPF Makefile Wizard**

**TPF Makefile Details**  
Define TPF makefile details.

Program | ASM Source Segments | C Source Segments | CPP Source Segments | SQC Source Segments | External References | Advanced

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**

TPF System Application Build Options Build Overrides Advanced

Basic sub-system name: BSS

Sub-system Name:

TPF source

Add

Browse...

Remove

Up

Down

Macro and copy file PDS

Add

Remove

Up

Down

# TPF Toolkit supports MakeTPF

- wizard to generate control files

**TPF Make Build List**

Use an external control file

External control file:

For example, /home/path/external.cntl

Use this project's control file

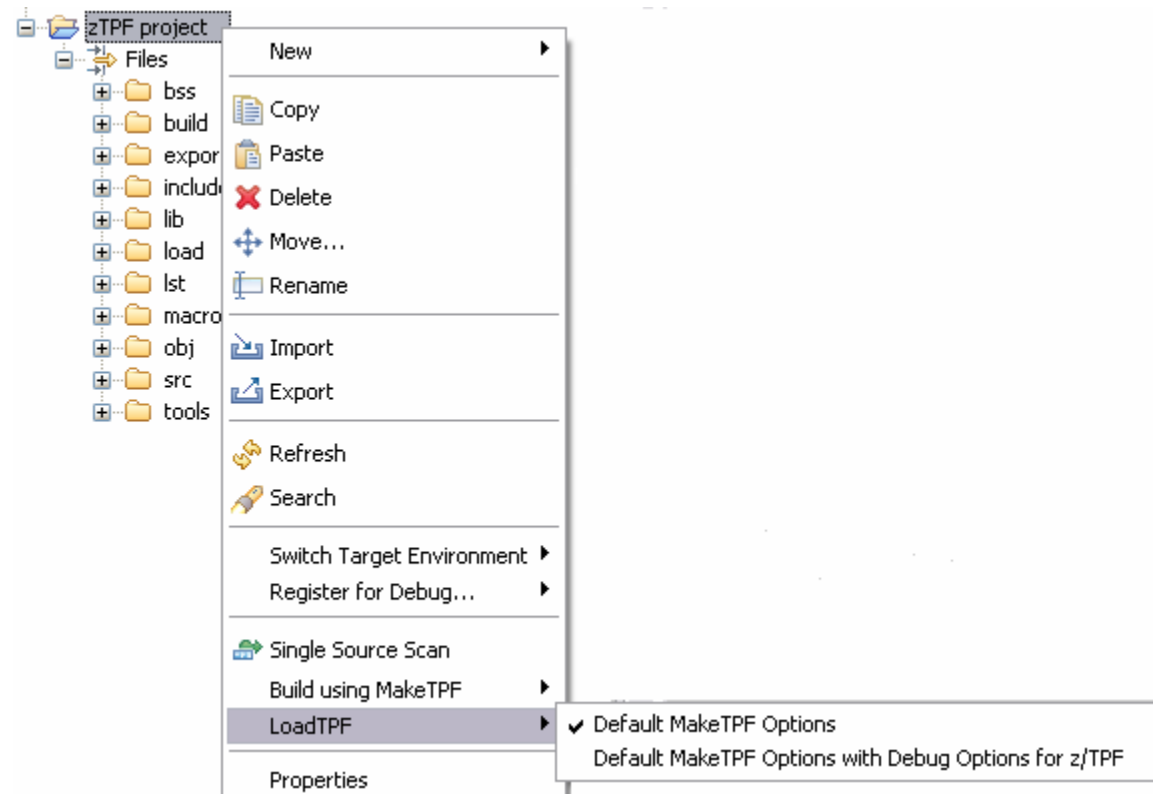
**Build List**

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

IBM is a trademark of International Business Machines Corporation in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries

Other company, product, or service names may be trademarks or service marks of others.

### Notes

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.