IBM TPF Toolkit 4.6.0

Matt Gritter
Software Engineer

GA – March 22, 2018

Contents

Overall Development Goals

The team's overall goal for this release was to provide a **simplified user experience** with a focus on **standard Eclipse solutions** following a development pattern of **continuous integration**.

Installing TPF Toolkit

Eclipse Installer, powered by Oomph, is the recommended solution for automating installing and update Eclipse to an enterprise community.

By customizing the installer and creating setup artifacts, a greater level of control is available over product deployment compared with IBM Installation Manager.

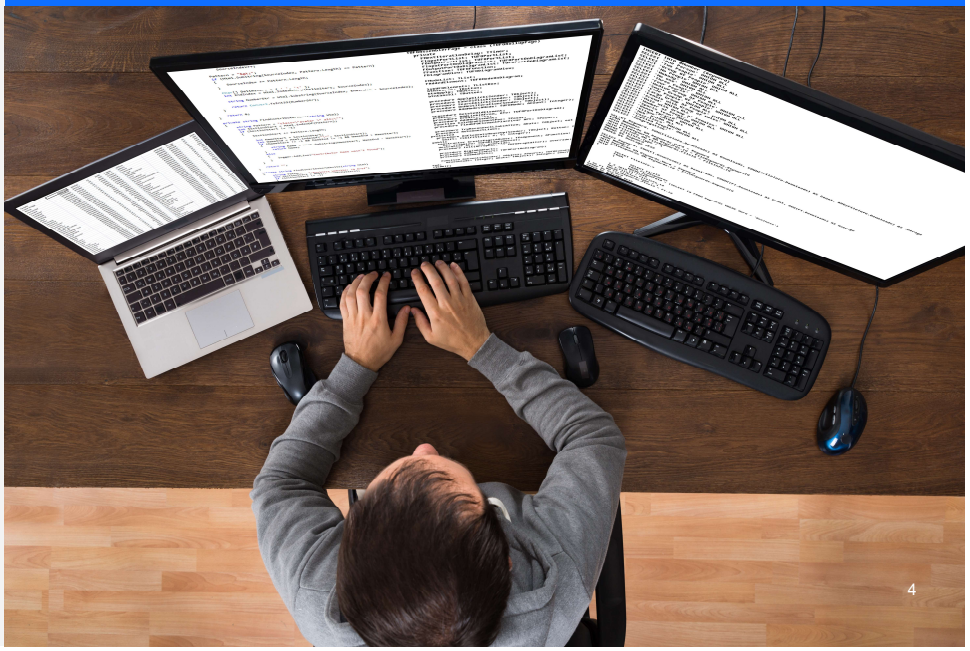
There are additional Oomph tools for Eclipse that will allow recording preferences and adding them to the setup artifacts.

Updates can be configured to occur automatically during startup.

Full product install including customization and third party tools with a single installation action

VS

Installing Installation Manager, TPF Toolkit, and additional features in three or more installation actions



Project templates

Extensible functionality provided by the Eclipse C/C++ Development Tools (CDT) project

Project templates allow administrators to provide custom pages and controls to the CDT New Project wizard.

Input from those pages can be substituted into files or used in custom processes that are executed as part of project creation.

Administrators can create folders, files and perform other actions all as part of project creation.

Eclipse standard solution vs. writing scripts and Menu Manager actions

After finishing the new project wizard, projects are ready to go, no additional configuration.

Provide support for new developers with a “Hello World” project unique to your code style and conventions.

Synchronized projects

Performance benefits on the remote system

Better merge handling

Functionality provided by the Eclipse Parallel Tools Platform (PTP) project.

Requires Git on the remote development environment vs. JVM and the Remote System Explorer DataStore server.

Ability to filter out build artifacts or other files - globally or per project

Gain the benefits of source files being local in the workspace

Synchronized projects store source files in the local Eclipse workspace and in a folder on the remote development environment. Local and remote changes are pushed between the two environments automatically or on demand.

Integrating MakeTPF

Many of the capabilities of the Eclipse CDT project relies on being able to resolve include paths outside of the project area.

With PTP and CDT integrated, remote paths are able to be resolved.

Normal CDT usage requires adding in paths one by one in the project's property page.

The Toolkit team created a command that will resolve the include paths based on maketpf.cfg and populate the property for that project. Runs against the project or a program makefile.

Command requires the cmd target in the project makefile

Building projects

New build infrastructure relies on a project makefile using the GNU make syntax

Each template project should contribute build targets for the common actions an end user would take on the remote system i.e. build, load, etc…

Variety of ways to invoke build targets provided in the project makefile

End users can add / replace build targets for advanced usage scenarios

Create build targets that call multiple other build targets for convenience

Change environment variables for particular targets

- Single click to build programs, generate and ftp loadset
- Single click to build programs for multiple environments

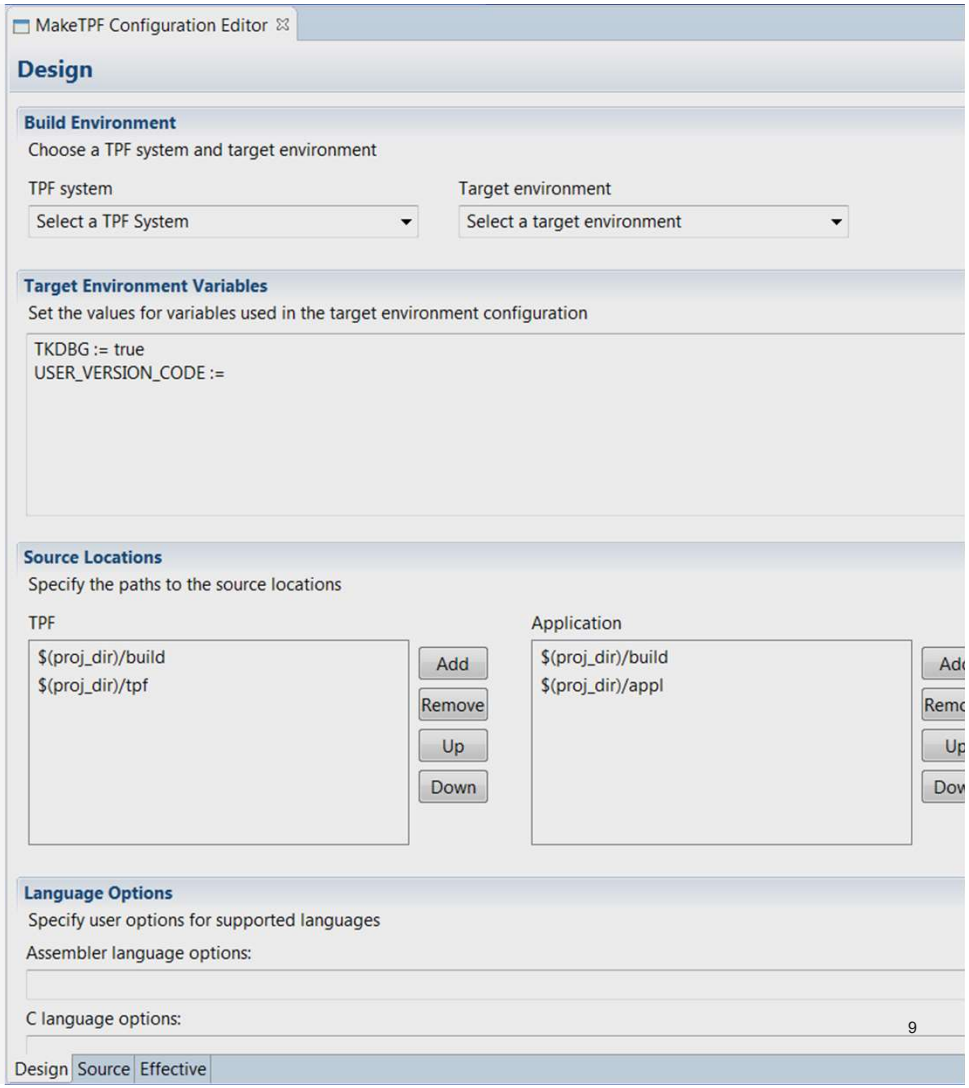
MakeTPF configuration editor

The MakeTPF configuration file editor has three pages: design, source, effective.

The design page provides controls for the most commonly changed variables.

The source page displays the full source of the file for advanced use cases.

The effective page merges the values specified in the configuration file and the values defined in the target environment configuration file and displays them in a list for the current build configuration for the project



Working with TPF systems

Majority of the TPF system functionality was tied to the Remote System Explorer in Toolkit 4.2.

The user experience creating and registering debug and code coverage session was frustrating.

For Toolkit 4.6, debug and code coverage sessions are not tied to a particular TPF system and are associated with their own views.

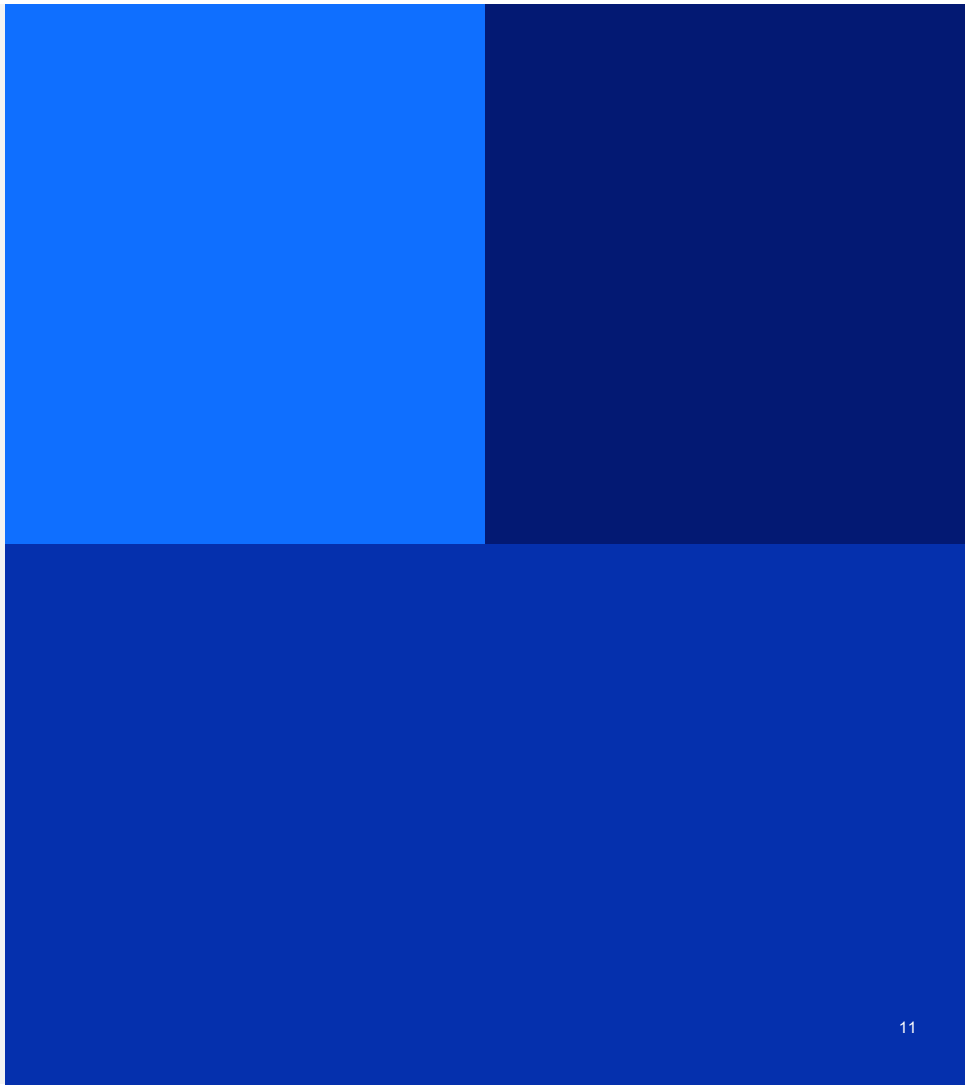
Toolkit administrators can contribute default TPF systems allows for easy addition of shared systems.

Toolkit Documentation in IBM Knowledge Center

The Toolkit and TPF documentation are in a common location in the IBM Knowledge Center.

The ability to link between TPF topics and Toolkit topics.

No more “search for this keyword” or “locate this topic” for cross-product features.

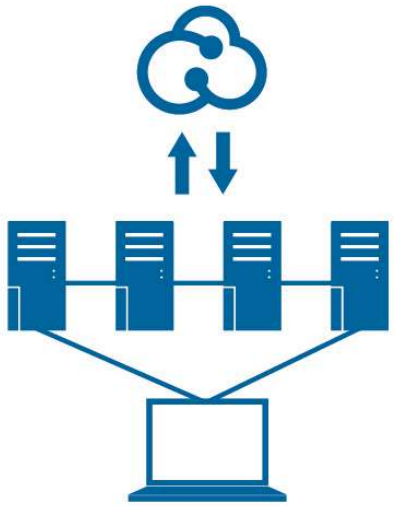


XML Deployment Descriptors

The XML deployment descriptors for business events, Java and MongoDB are now created and modified using the standard Eclipse XML editor.

The schemas describing the deployment descriptor content are stored in the XML catalog using extension points from IBM delivered plug-ins.

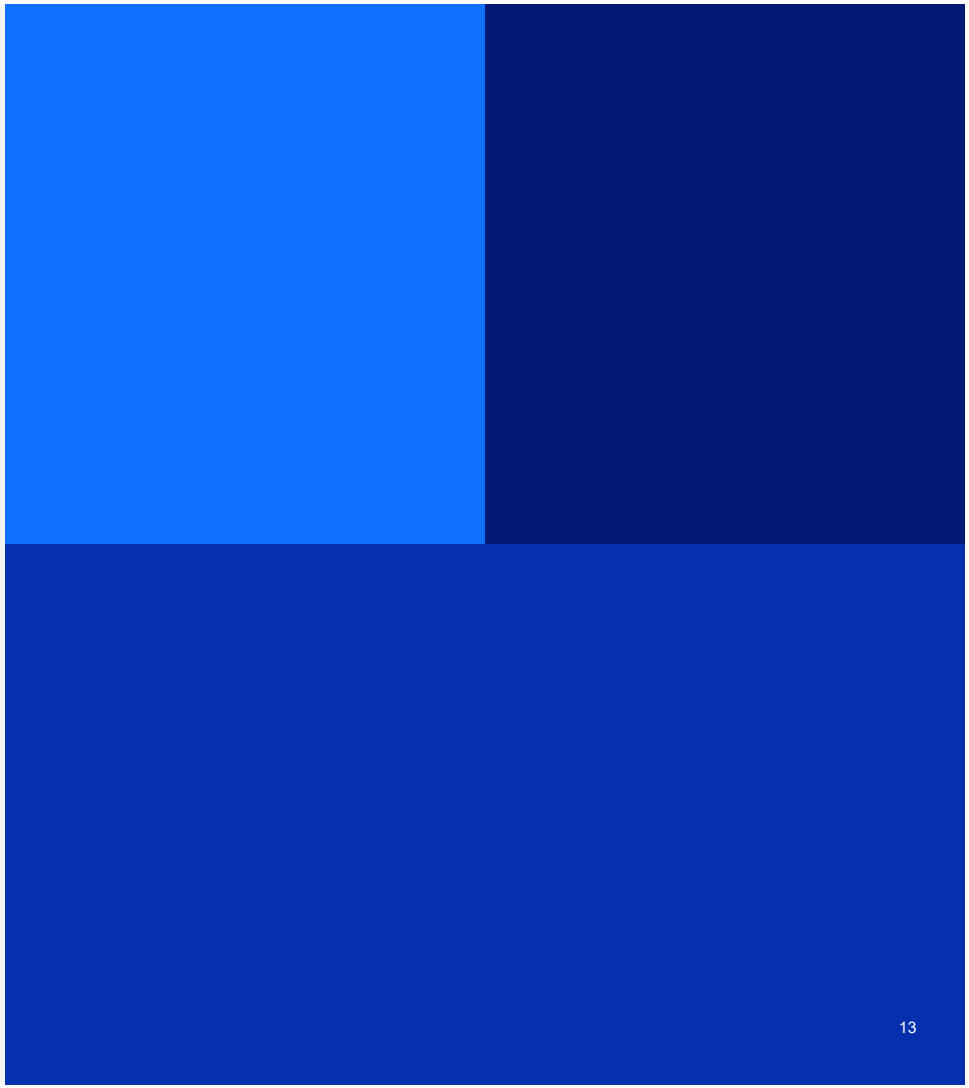
This allows for on-the-spot validation and a common editing experience for deployment descriptors.



Dynamic Tracing

The TPF Toolkit plug-ins are integrated with the Eclipse Tracing platform allowing users to dynamically turn tracing on and off as well as specifying the path to write the file.

No longer have to update a bat file and restart TPF Toolkit to collect trace information for the TPF developed plug-ins.



Multiple Platforms

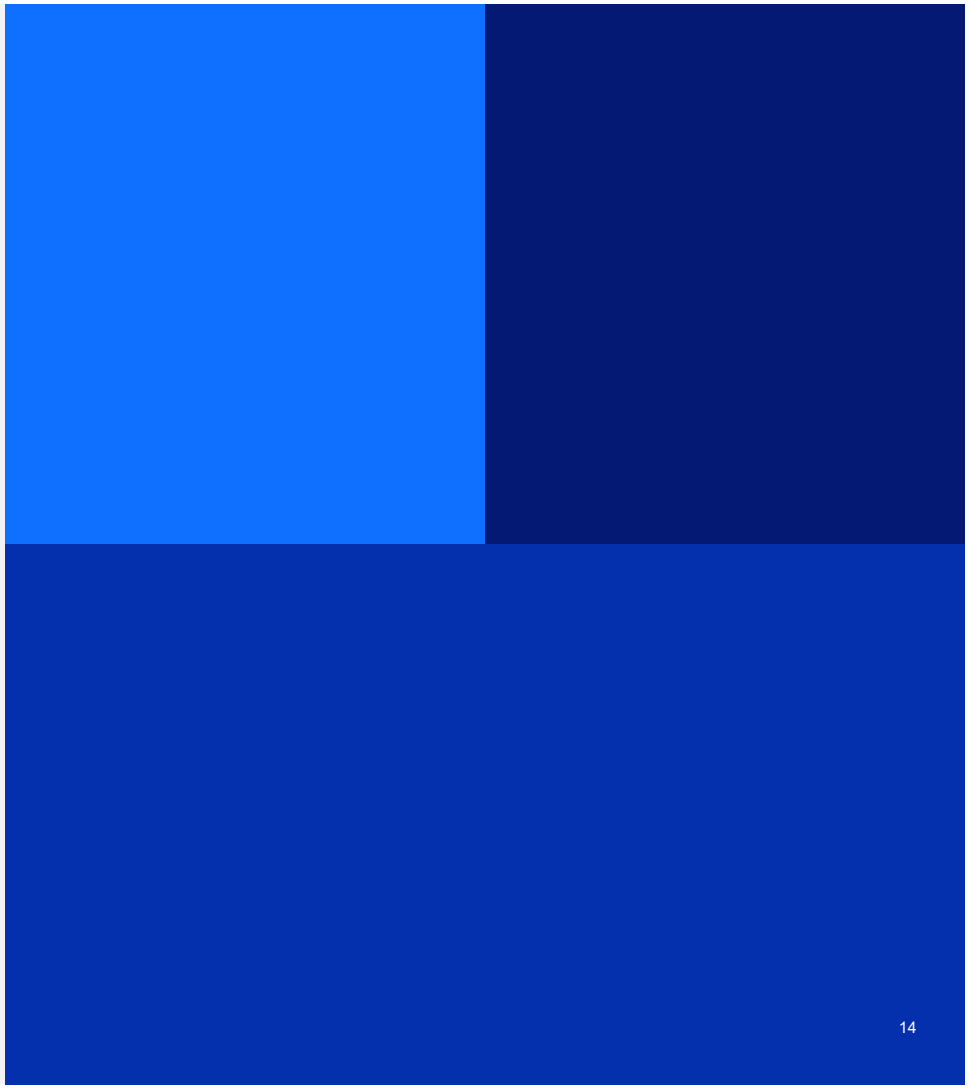
TPF Toolkit can run on the Windows and **Linux x86** platforms that are supported and tested for Eclipse Neon.

Windows 7 and 10

Red Hat Enterprise Linux 6 and 7

SUSE Linux Enterprise Server 12 SP1

Support for Linux environments can help ease costs if Windows is only being used for TPF Toolkit today.



Continuous Integration for better quality from the team

- A team member delivers a change set to the 4.6 stream.
- A nightly Jenkins jobs invokes Maven to build the product from the 4.6 development stream.
  - The Toolkit product build includes running JUnit tests for each module
  - The static analysis tooling for Java code (FindBugs, SonarQube) is also run as part of the product build.
  - Team responds to any blocker, critical, major severity issues before delivery. Lower severity issues are handled on a case by case basis, most are Java style conventions, etc.
- If the product build job is successful, the down-stream integration test Jenkins jobs are started on VMs.
  - The product archive files are downloaded from the build results and the UI integration test cases are run on Windows 10 and Red Hat Enterprise Linux 7
- If the integration test jobs are successful, another down-stream job is started to deploy the product when a beta period is active.

Continuous Integration by the numbers for the Toolkit product

- Toolkit product build for the 4.2 stream takes 3 hours to complete without regression tests.
  - Two large and brittle UI function test cases that are not integrated with the build
  - No static analysis on the Java source code integrated with the build
- Toolkit product build for the 4.6 stream takes 15 minutes with integrated JUnit tests
  - UI regression test suite takes an additional 10-17 minutes to run per execution.
  - 150 JUnit and UI regression test cases and growing

Continuous Deployment by the team

- Parallel but not duplicate development – two streams for the Toolkit 4.6 product: a development stream and a release stream
- Once a feature is developed and tested in the development stream, a release engineer delivers the change sets from the development stream to the release stream. These change sets include any regression test cases that were developed for the feature.
- A release build is run from Jenkins including the regression test cases.
- The release stream is always in a ready to publish state.
- Post-release, the release stream is updated using Maven plug-ins to increment the version numbers and any dependencies that are affected.
- The release engineer then delivers the change sets from the release stream to the development stream and the developers resolve any conflicts.

Interested in helping setting priorities for the Toolkit team?

Become a sponsor user today!

Ask Josh Wisniewski or Matt Gritter for more information.

Bi-monthly playbacks with designs, demos, discussions.

Web resources

Eclipse Oomph	www.eclipse.org/oomph
Eclipse CDT	www.eclipse.org/cdt
Eclipse PTP	www.eclipse.org/ptp
Eclipse Help	help.eclipse.org
z/TPF Knowledge Center	www.ibm.com/support/knowledgecenter/SSB23S

* now includes TPF Toolkit documentation

TPF Toolkit Taskforce

For a more in depth discussion of these features and additional topics, we will hosting a playback session during the TPF Toolkit taskforce, tomorrow morning at 8AM.

Thank you

Matt Gritter
Software Engineer
—
mrgritte@us.ibm.com
ibm.com

Trademarks

Notes

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

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.