

News in RSA-RTE 10.2

updated for sprint 2018.40

Mattias Mohlin, October 2018



Overview

- **Now based on Eclipse Oxygen.3 (4.7.3)**
- **Contains everything from RSARTE 10.1 and also additional features and bug fixes**
 - See the What's New presentation for RSARTE 10.1 to learn about the new features that are also present in version 10.1



IBM Rational® Software Architect RealTime Edition

Version: 10.2.0.v20180926_1406

Release: 2018.40

(c) Copyright IBM Corporation 2004, 2016. All rights reserved.

(c) Copyright HCL Technologies Ltd. 2016, 2018. All rights reserved.

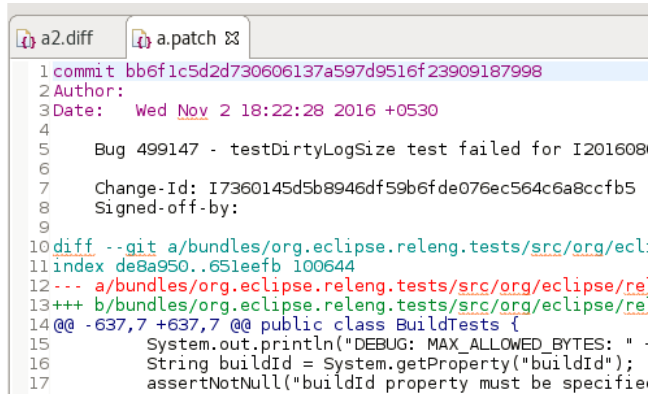
Visit

https://www.ibm.com/developerworks/community/wikis/home?lang=en#/wiki/W0c4a14ff363e_436c_9962_2254bb5cbc60/page/Rational%20Software%20Architect%20RealTime%20Edition%20Wiki



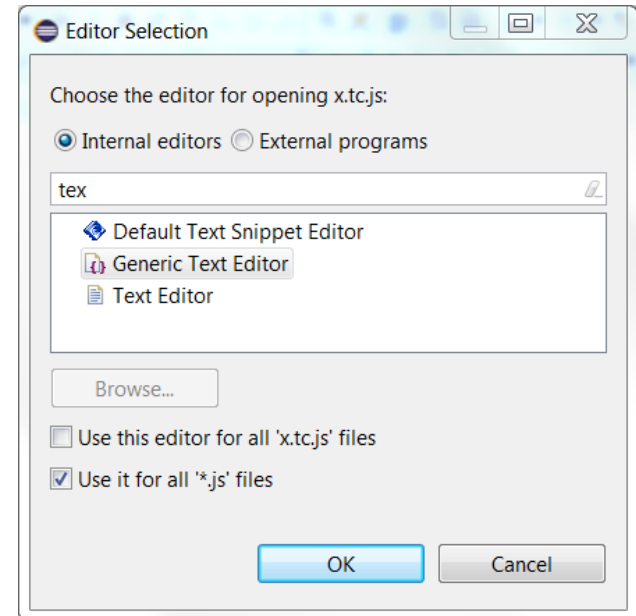
Eclipse 4.7.3 (Oxygen)

- New text editor ("Generic Text Editor")
 - Has some general improvements, such as support for syntax highlighting of .patch and .diff files



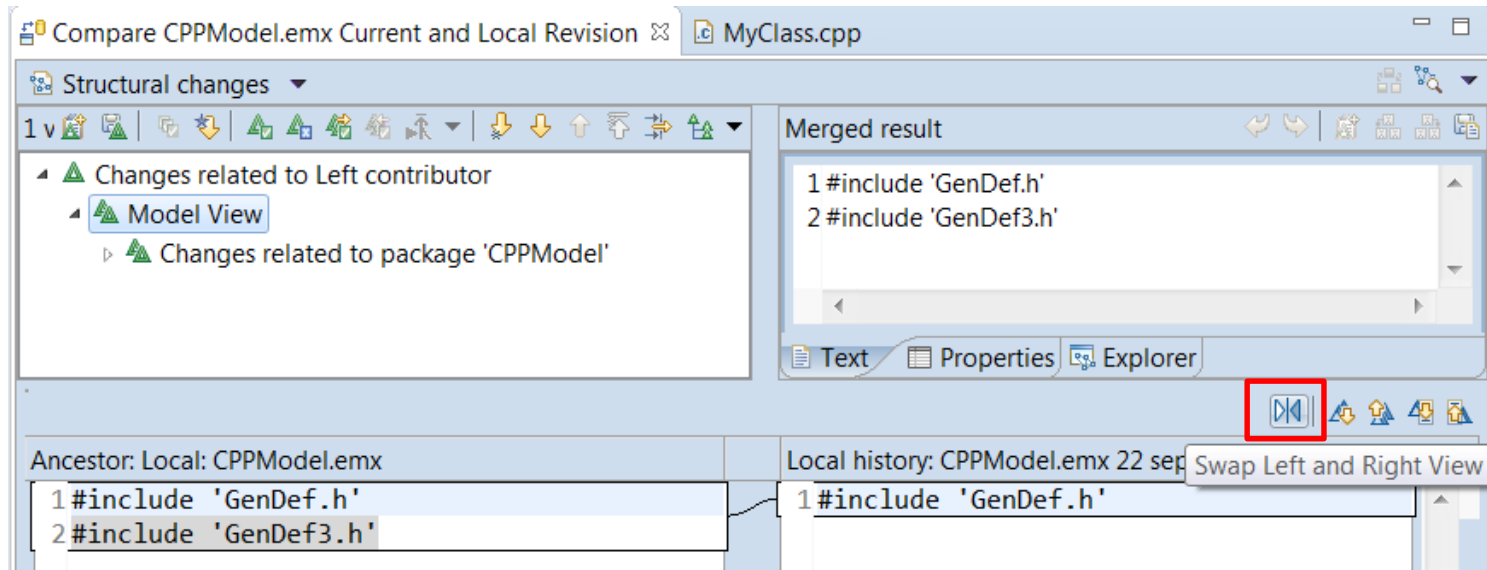
```
a2.diff a.patch ✕
1 commit bb6f1c5d2d730606137a597d9516f23909187998
2 Author:
3 Date: Wed Nov 2 18:22:28 2016 +0530
4
5 Bug 499147 - testDirtyLogSize test failed for I201608
6
7 Change-Id: I7360145d5b8946df59b6fde076ec564c6a8ccfb5
8 Signed-off-by:
9
10 diff --git a/bundles/org.eclipse.reneng.tests/src/org/ecl:
11 index de8a950..651eeeb 100644
12 --- a/bundles/org.eclipse.reneng.tests/src/org/eclipse/re:
13 +++ b/bundles/org.eclipse.reneng.tests/src/org/eclipse/re:
14 @@ -637,7 +637,7 @@ public class BuildTests {
15      System.out.println("DEBUG: MAX_ALLOWED_BYTES: "
16      String buildId = System.getProperty("buildId");
17      assertNotNull("buildId property must be specifie
```

- Easier to associate files with special extensions to an appropriate editor
 - Select the file and do *Open with – Other...*
 - Specify an internal Eclipse editor, or an external program
 - For example, set .tc.js files to be opened with your favorite text editor



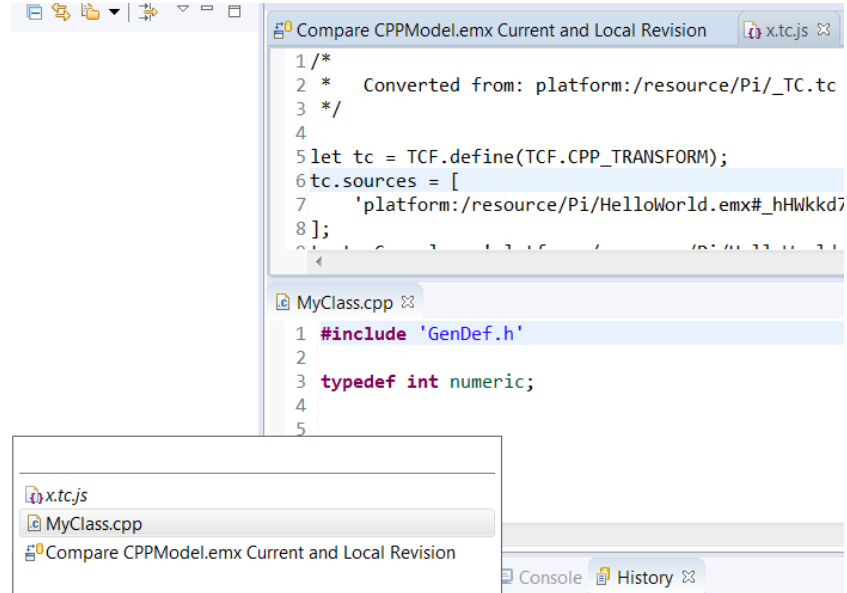
Eclipse 4.7.3 (Oxygen)

- Swapping left and right side when doing textual compare/merge
 - Useful if selecting files so they were compared in the "wrong" order
 - Available for all text files, and also when doing compare/merge on a code snippet within a model



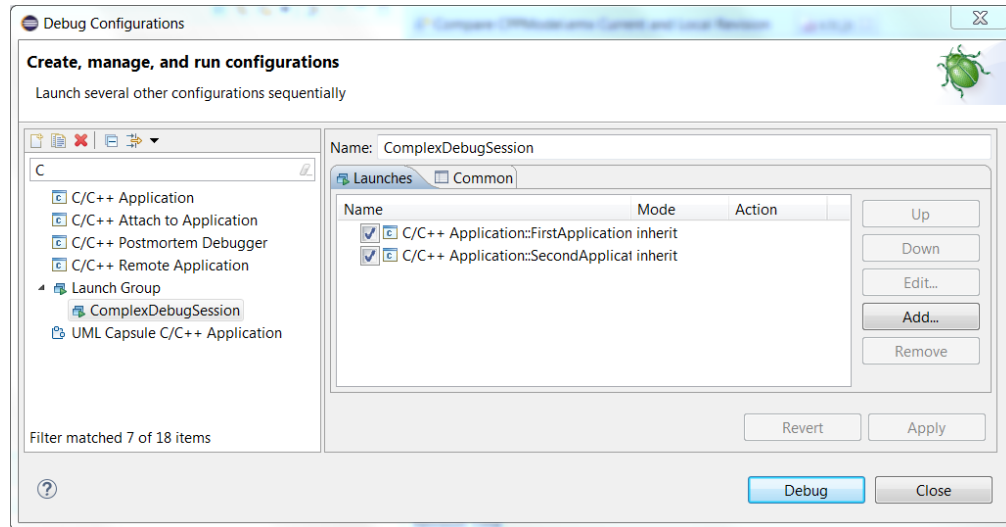
Eclipse 4.7.3 (Oxygen)

- It's now possible to switch between open editors using Ctrl+E even when the editor area has been split to show more than one editor at the same time
 - Especially useful for users with big screens who often show more than one editor at the same time
 - Tooltip with open editors now appear in the middle of the screen to make it easier to notice
 - Navigation to subsequent editors can now be done using Ctrl+E instead of having to use arrow keys or mouse



Eclipse 4.7.3 (Oxygen)

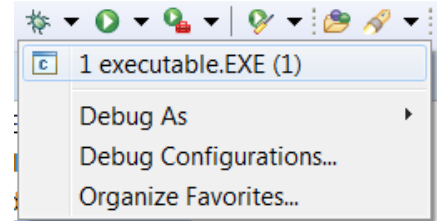
- Support for launch groups
 - A new type of launch configuration that can contain other launch configurations
 - Makes it easier to debug applications that require multiple executables to be launched
 - Possible to customize the rules for when and how the contained launch configurations should be launched
 - N.B. CDT previously provided a similar launch group feature, and it is now deprecated



Eclipse 4.7.3 (Oxygen)

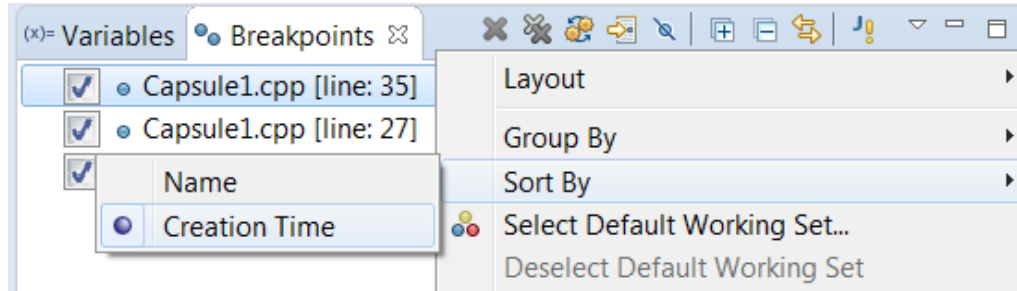
■ Terminate and Relaunch

- The default Eclipse behavior is to always launch a new debug session, when launching from the history in the Debug and Run button menus.
- Now you can press Shift when launching from the history to automatically terminate the previous session before launching a new session
- A new preference *Run/Debug – Launching – Terminate and Relaunch while launching* can be set if you always prefer this behavior (without pressing Shift)



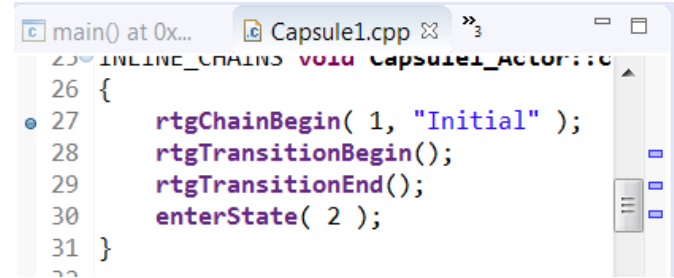
■ Sorting breakpoints by creation date

- Useful when there are many breakpoints and you want to see the newest one on top

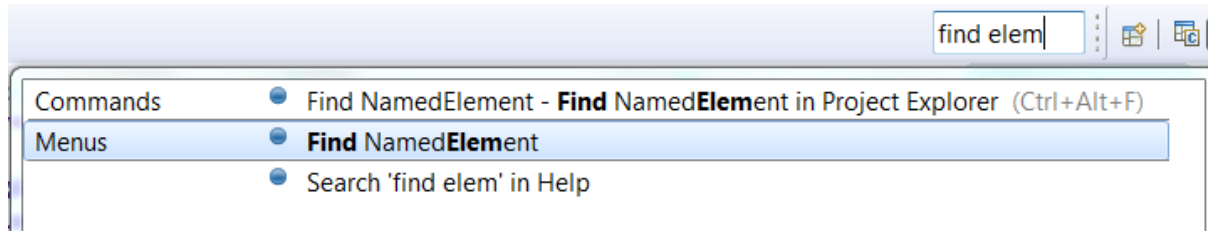


Eclipse 4.7.3 (Oxygen)

- Breakpoints now visible in the overview ruler
 - Helps finding the breakpoints in large files
- Smarter Quick Access search
 - Now supports space separated strings
 - Useful if you don't remember the command name exactly
 - Wildcards (* and ?) can also be used
 - Also possible now to use Quick Access for searching in the Help documentation
 - Other minor improvements (e.g. showing command icons) also make this feature more user-friendly

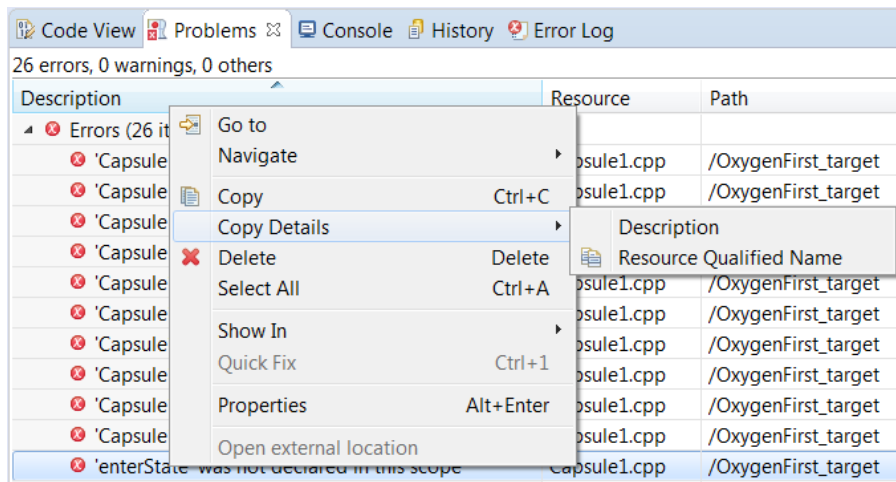


```
25 inline void Capsule1Actor::rtgChainBegin( int state, const char* name )
26 {
27     rtgChainBegin( 1, "Initial" );
28     rtgTransitionBegin();
29     rtgTransitionEnd();
30     enterState( 2 );
31 }
```



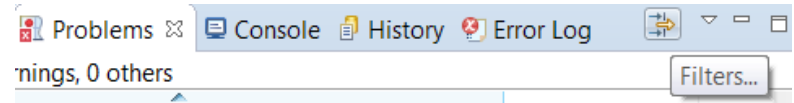
Eclipse 4.7.3 (Oxygen)

- Possible to hide the status bar
 - A new command *Window – Appearance – Hide (Show) Status Bar* can be used
- (Linux only) Left/Right arrow keys for collapsing/expanding current tree node
 - Same behavior as on Windows
 - Works on GTK+ 3.6 and later
- Easier to copy parts of an entry in the Problems or Tasks view
 - Copy Details in the context menu can be used for copying either only the description text or the resource name

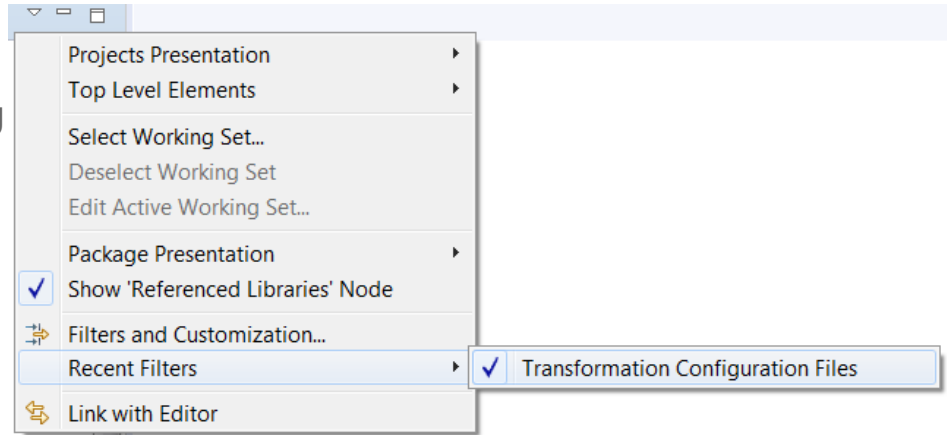


Eclipse 4.7.3 (Oxygen)

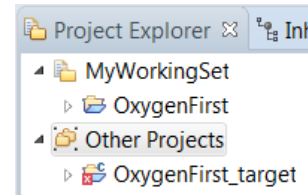
- Easier to filter the Problems and Tasks views
 - Filters can be applied using a toolbar button
 - The Filters dialog itself has also been simplified



- Easier to filter the Project Explorer
 - A new context submenu for setting/unsetting recent filters
 - Note: The command "Customize view" is now called "Filters and Customization"

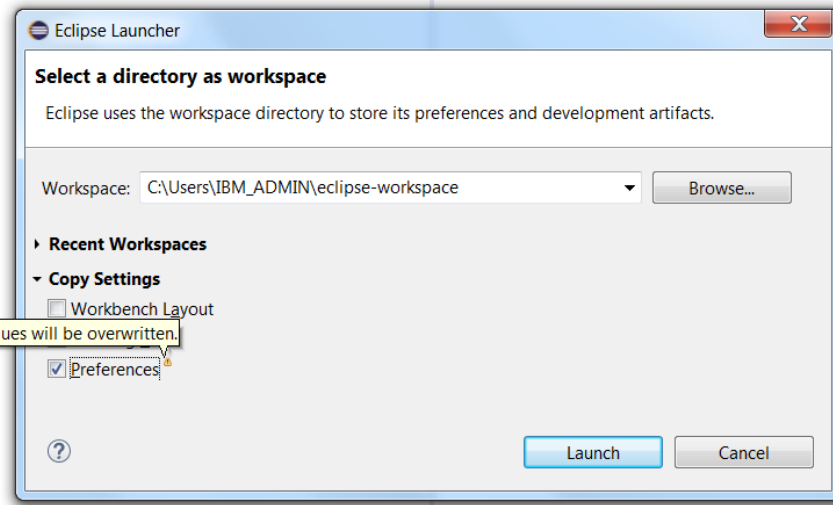


- Showing projects not in the currently active working sets
 - An "Other Projects" group can be used for showing such projects when working sets are the top level elements



Eclipse 4.7.3 (Oxygen)

- Copying workspace preferences
 - Can now be done when switching to a new or existing workspace
- Always run in background
 - The preference *General – Always run in background* is now enabled by default. Disable it if you prefer the old behavior of seeing a progress dialog for long-running operations.
- More external web browsers supported on Linux
 - Now support for Firefox, Chrome, Chromium, Epiphany/Gnome Web, Konqueror
- For more information about Eclipse improvements see
 - News in Eclipse 4.7 (Oxygen) <http://www.eclipse.org/eclipse/news/4.7/platform.php>



CDT 9.4 (included as part of Eclipse Oxygen.3)

- **Open Declaration**
 - This command has been improved for several navigation scenarios. For example, it's now possible to navigate from a class template to its forward declaration.
- **Content Assist**
 - Recognition of the pattern **&ClassName::** to include also non-static member functions as proposals
 - Now supports the case when include files do not have traditional file extensions (.h or .hpp)
 - More information is now printed in the hint tooltip for function parameters (full function signature)
- **Comment Folding**
 - Documentation comments (///) are now better supported w.r.t folding in the CDT editor
- **Performance Improvements**
 - Build Console is now up to 25 times faster when performing large number of printouts to the console
 - UI responsiveness has improved by reducing the number of threads used
 - Now the build time when using CDT is comparable to when building from command-line



CDT 9.4

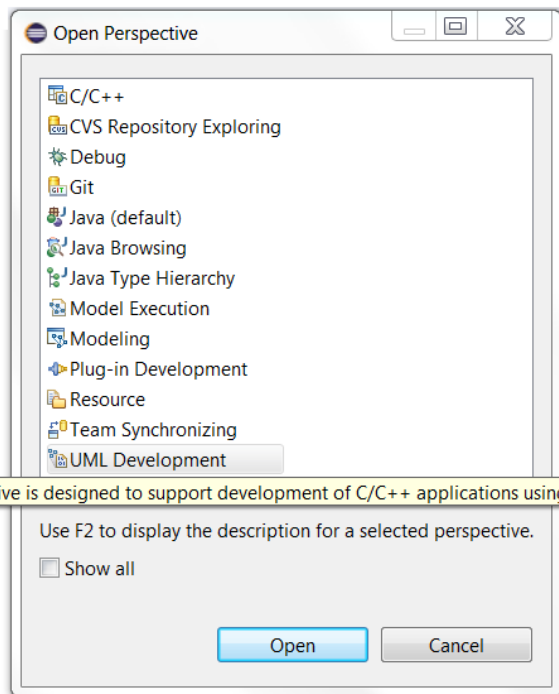
- Source Not Found Editor
 - A new preference in C/C++ - Debug allows to control when this editor appears
- Quick Fixes
 - More Quick Fixes are now available for many build errors generated by gcc

*For more information about news in CDT 9.4
see <https://wiki.eclipse.org/CDT/User/NewIn94>*



Perspective Improvements

- Eclipse Oxygen now supports showing perspective descriptions in the Open Perspective dialog



- The perspective description is shown by pressing F2
- Helps in particular new RSARTE users in learning the tool
- All perspectives provided by RSARTE now have a description
- **Note:** The perspective "Classic Modeling" has been removed as it was considered confusing to have two different perspectives for the purpose of modeling. The "Classic Modeling" perspective was intended for users used to RSARTE versions earlier than 8.0 and didn't provide any real value for users used to more modern versions of RSARTE. If needed, it is always possible to create a customized version of the Modeling perspective.



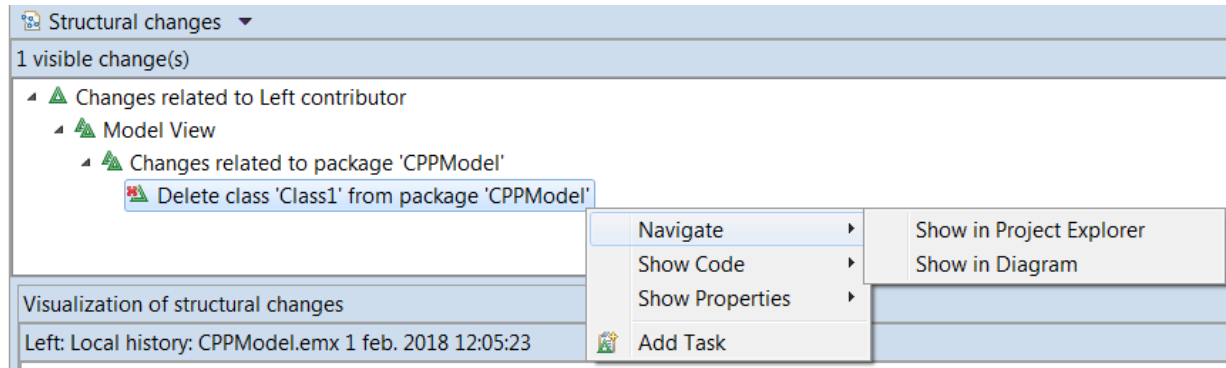
Read-Only Installation

- It's now possible to install RSARTE into a read-only Eclipse installation
 - RSARTE specific tools need no longer be present in an **rsa_rt** folder in the installation
- Note that the TargetRTS and Connexis features still cannot be installed into a read-only Eclipse installation
 - The workaround is to install those features in a different (writable) Eclipse installation and then copy the folders **rsa_rt/C++** and/or **rsa_rt/Connexis** to some shared folder **rsa_rt** and use it from the read-only RSARTE installation.



Compare/Merge Navigation Commands

- New command for navigating directly to a diagram from a change
 - Previously it was necessary to first navigate to the Project Explorer and from there to the diagram
- The command for navigating to the Project Explorer was renamed to "Show in Project Explorer" to be consistent with other similar navigation commands



Compare In-Memory Model with Local File

- A new command makes it possible to see how the model has been changed since it was last saved
 - Use it on modified (“dirty”) models to review changes before saving them
 - For example useful in case something gets modified when you didn’t expect it to



Resolve Conflicts with Both Contributor Versions

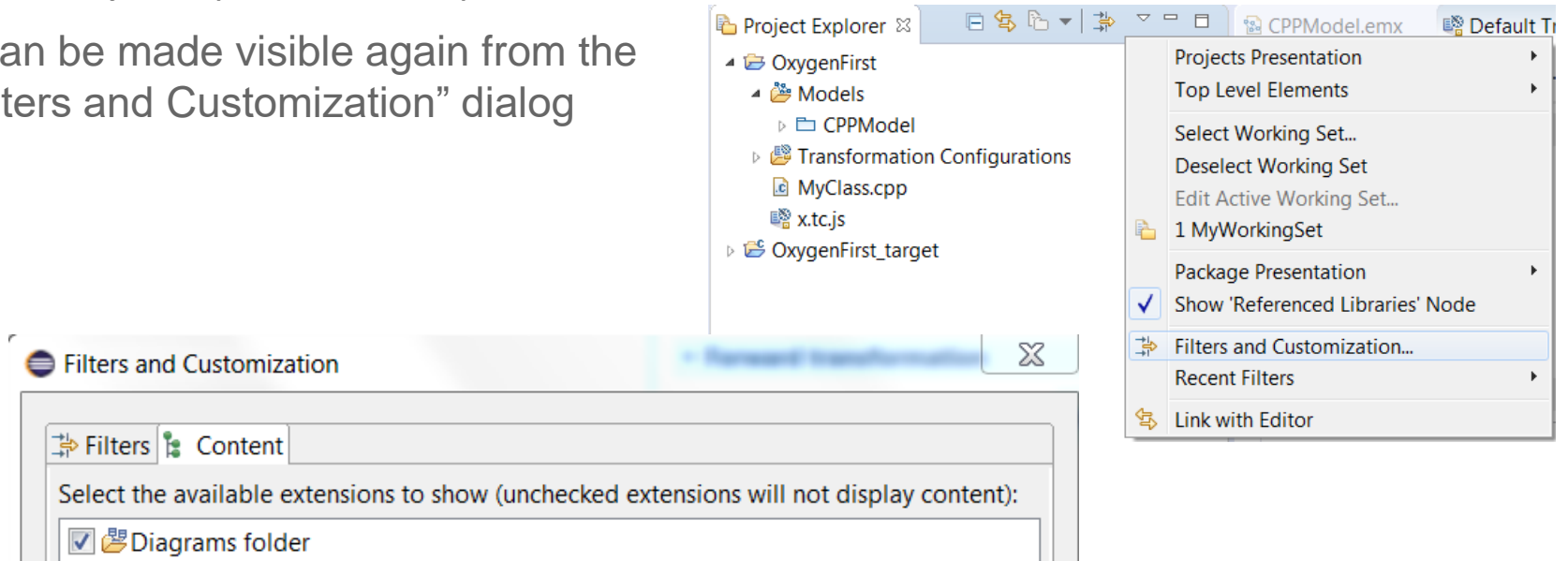
- Textual merge conflicts can now be resolved by applying the changes from both the left and right contributors
 - Useful when both changes make sense and should be included in the merged version
 - Previously it was necessary to manually copy/paste from the contributors to create a merged version that contained the changes from both contributors. Now a toolbar button can be pressed instead.

```
Conflict: Modify 'Header Preface' of capsule 'Capsule1'. Current state: Unresolved.
1 #include <string.h>
2 #include "IncludeFile1.h"
Added branch include - a06da37 3 <<<<<<< Added branch include - a06da37
4 #include "branch.h"
5 =====
add master include - 17f5c79 6 #include "master.h"
7 >>>>>>> add master include - 17f5c79
8
```



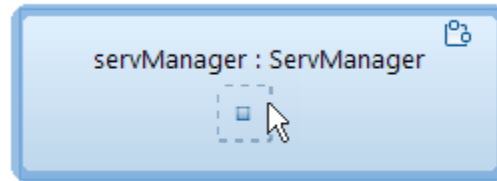
Hidden Diagrams Folder in Project Explorer

- The Diagrams folder is now by default not shown in the Project Explorer
 - This folder is not very useful when working with big models
 - The Project Explorer looks simpler without it
- It can be made visible again from the "Filters and Customization" dialog



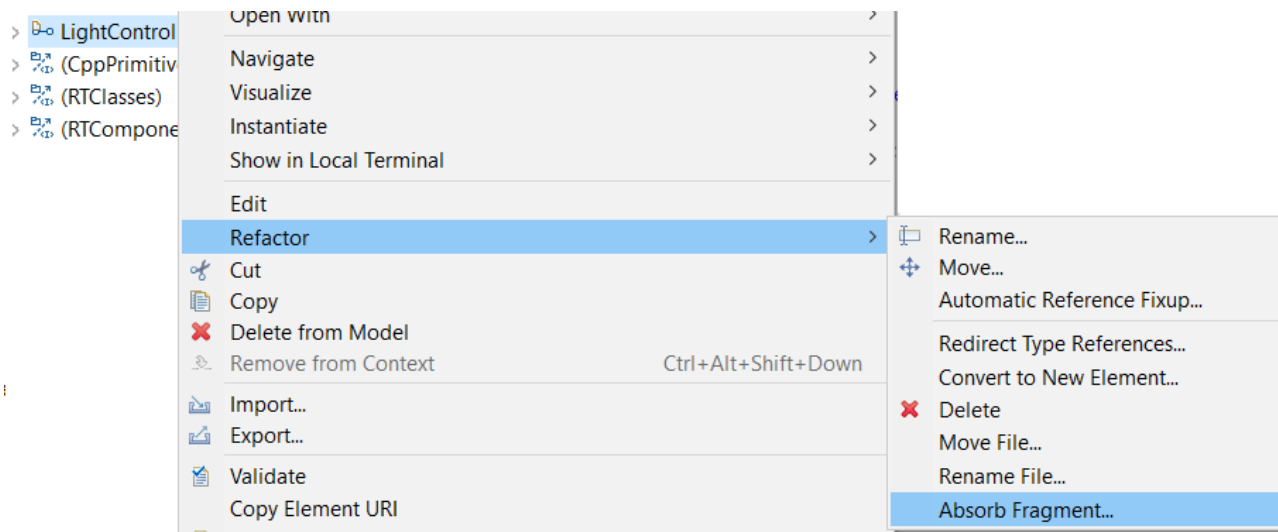
Port Creation on Capsule Parts

- Now service ports are always created when dropping a port from the palette onto a capsule part
 - Previously it was necessary to drop the port on the capsule part frame to achieve this
 - Accidentally dropping it inside the capsule part would create a non-service port, which therefore was not shown on the capsule part. This behavior was confusing and could lead to creation of unwanted ports.
 - To create a non-service port, just select the created port and unmark the "Service" checkbox in the Properties view



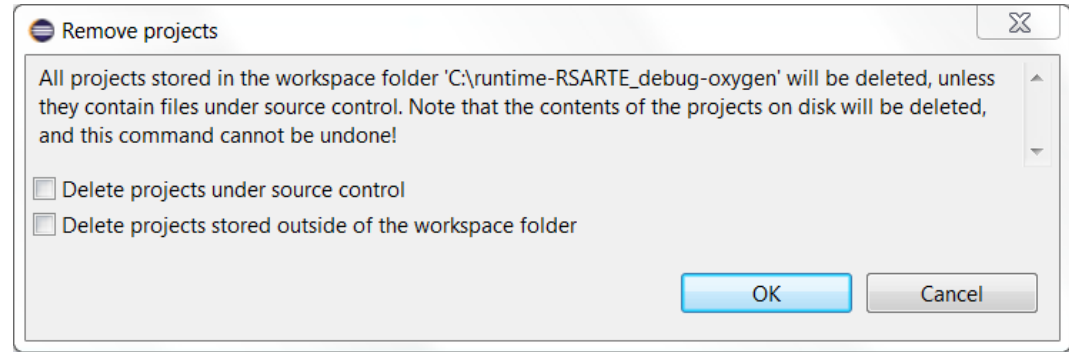
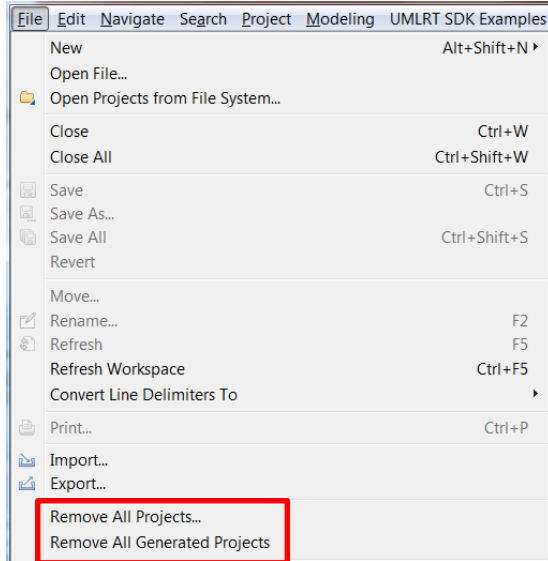
Protocols in Fragment Files

- Protocols can now be stored in fragment files
 - Now behaves in the same way as classes, capsules, packages etc.
 - A protocol fragment can be absorbed using the **Absorb Fragment** context menu command



Removing Projects from Workspace

- New commands in the File menu make it easy to remove all projects from the workspace



- These commands are more convenient than deleting the projects since they will not trigger any refactorings (which otherwise can make the removal unnecessary slow)
- By default projects under source control and projects stored outside the workspace folder will not be deleted
- The command "Remove All Generated Projects" is useful for quickly removing all generated CDT projects from the workspace
- Note! These commands are not undoable!



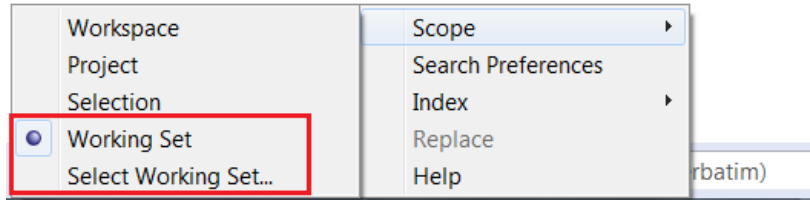
Search Improvements (1/3)

- Search (and replace) now supports the new TC file format (*.tcjs)
 - The files are indexed and search can therefore find TC settings defined in such files
 - Navigation to TC files shown in the virtual "TCJS Transformation Configurations" is supported
- Backslashes are now only interpreted as escape characters when immediately followed by a wildcard (* or ?) or another backslash
 - Allows for example to search for Window-style paths in TC properties
- A new preference *Team – External Projects – Ignore pattern* allows to exclude from the map file projects that do not exist
 - Previously missing projects caused exceptions, but now warnings are reported instead (and by using the new preference the missing projects can be completely ignored)
 - The new preference can be useful if a common map file is used in an organization, but where some users may not have access to all listed projects
- A new preference *UML Development – Search – Use External Projects scope by default*
 - Can be set to search in external projects by default

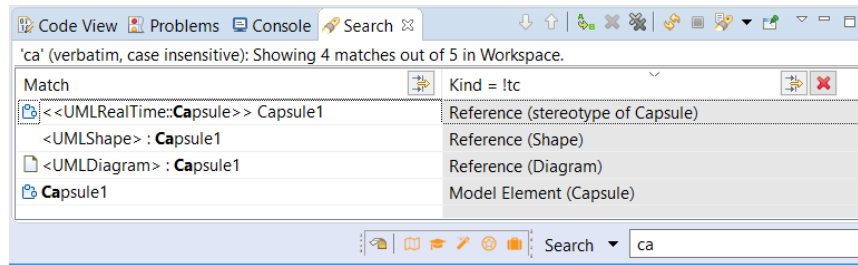


Search Improvements (2/3)

- Support for searching in working sets from search field
 - Synchronized with the corresponding setting in the Model Search dialog

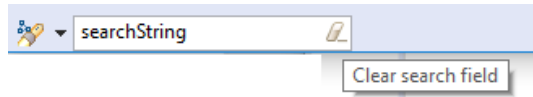


- Support for negative search patterns
 - You can now put an exclamation mark (!) in the beginning of a filter string for a column to negate the search filter for that column (i.e. so that it matches everything except the string that follows)



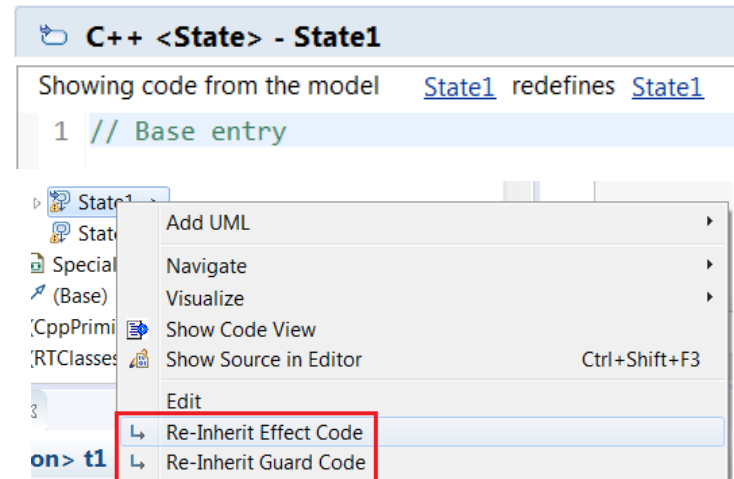
Search Improvements (3/3)

- The Search view is now filtered after the query has run against the index
 - This means that new matches can appear as long as the total match count is below the specified limit
 - This way of filtering is more intuitive, but is also more time-consuming (all matches returned from the index have to be compared against the specified filter)
- Search is now blocked while the model is being indexed
 - The search can be performed, but will not run until indexing has completed
 - Avoids unexpected search results by preventing searches against an incomplete index
- The Search field can be cleared by means of pressing a Clear button



Redefined Code Snippets

- An inherited code snippet can now be redefined by simply removing all code in the code view or code editor
 - Previously this would edit the base code snippet which was confusing
- The code view and code editor shows more clearly now when a code snippet is redefined
 - Correct icon is shown
 - A new hyperlink allows navigating to the base code snippet (both in Project Explorer and diagrams)
- The commands for re-inheriting (i.e. remove a redefinition) are now available in the Project Explorer context menu
 - Previously they were only available in diagrams



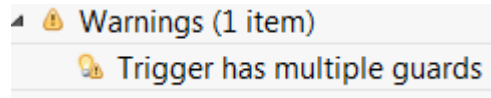
Improved Profile Migration Algorithm

- The algorithm for migrating applied profiles from one version to another was improved
 - The new algorithm ensures that IDs of applied stereotypes remain unchanged
 - This simplifies Compare/Merge of models that use different profile versions (there are now much fewer changes reported in this case)
 - Note: Profiles that were already migrated in the past are not affected by this improvement



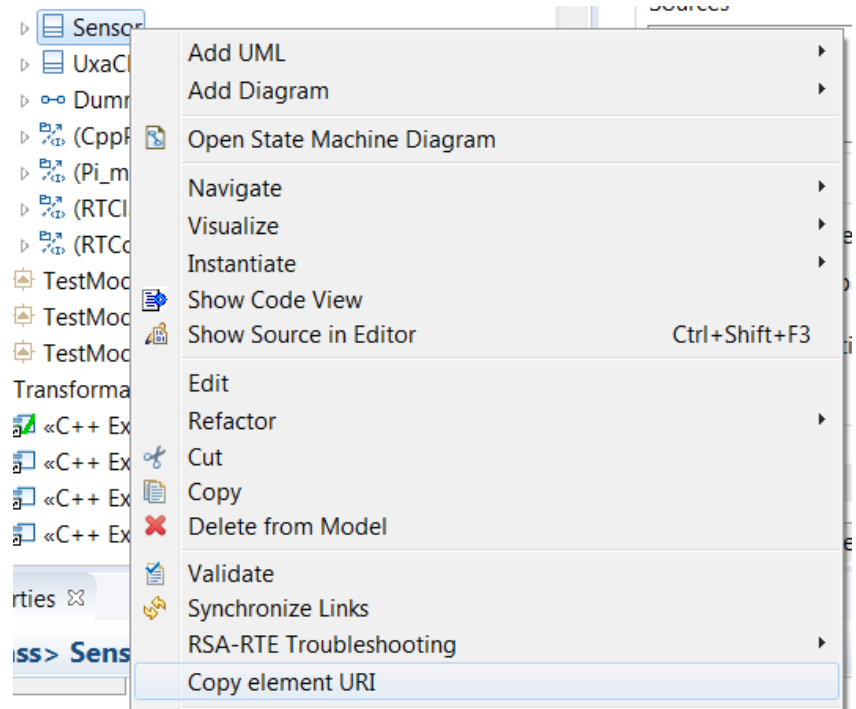
New Validation Rule

- Multiple guards for transition triggers are now detected
 - The reason for having duplicated guards could be merges done in older versions of RSARTE
 - Multiple trigger guards are supported by the C++ code generator, but is not well supported in the user interface, and is therefore usually unintentional
 - Duplicated trigger guards can cause problems for code-to-model synchronization and it's recommended to remove the duplicates
 - The validation rule runs "on-demand" (i.e. when performing the Validate command) and reports a warning if multiple guards are detected



Copy Element URI

- A command is now available for copying the URI of a file or element
 - Present in the context menu of files and model elements
 - This is useful in cases when you need the URI of an element (one example is when using the *Navigate - Navigate to URI* command)



Automated Code-to-Model Synchronization

- A new Ant task is available that makes it possible to automate the process of synchronizing changes from code to model
 - Can be run either from inside RSARTE or from command-line
 - In particular useful if generated code is changed frequently outside of RSARTE

```
<project name="CodeToModelSynch" default="synch">
  <target name="synch">
    <com.ibm.xtools.umltdt.rt.transform.cpp.codeSync
      transformConfig="/MyProject/HelloWorld.tc"
    />
  </target>
</project>
```



Model Compiler Improvements (1/4)

- The model compiler now allows usage of environment variables in the "Include file name" property for External C++ Library TCs
 - This avoids the need to hardcode the pathname of the include file that gets included in the unit header file for external libraries that are used
- RSARTE provides two new Ant tasks for generating the model compiler map and environment files
 - Then you can manually edit these files as necessary

```
<target name="createMapFile">  
  <com.ibm.xtools.umltdt.rt.transform.cpp.generateMapFiles mapFilePath="path_to_mapfile"/>  
</target>
```

```
<target name="createEnvFile">  
  <com.ibm.xtools.umltdt.rt.transform.cpp.generateMapFiles envFilePath="path_to_env_file"/>  
</target>
```



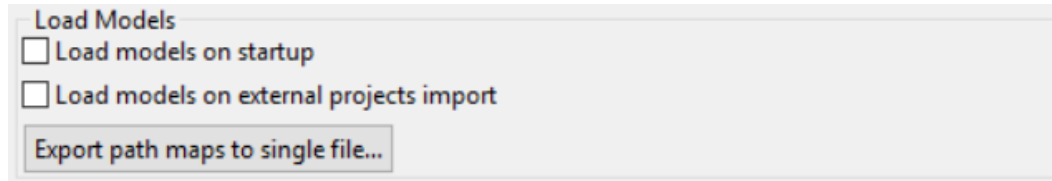
Model Compiler Improvements (2/4)

- **Faster and more accurate context sensitive library builds**
 - Achieved by skipping dependencies that represent forward declarations during analysis of source dependencies
- **Model Compiler Server**
 - The model compiler can now be started in a mode where it acts as a server, in order to serve multiple requests from the RSARTE user interface more efficiently
 - This avoids the overhead of launching the model compiler several times
 - Launch and termination of the model compiler server is fully automatic. From the user's point of view it can be seen as a pure performance optimization.
- **TC validation**
 - The model compiler now detects cycles in TC prerequisites and inheritance and terminates with a diagnostic message if such cycles are found
 - URIs to model elements (for example the top capsule) are now validated
- **Automatic save of models**
 - Models are now always saved before building with the model compiler (regardless of the preference *UML Development – Save affected files before running transformations*)



Model Compiler Improvements (3/4)

- Support for custom path maps when building from command-line
 - Pathmaps are useful to avoid hardcoded paths of libraries referenced from a model
 - Custom path maps can be defined in *Preferences – Modeling – Path Maps* (or programmatically using an extension point)
 - When running the model compiler from the command-line, you can now specify custom path maps using a new `--pathmap` option.
 - The argument of this option is a JAR file that can be created from the *UML Development* preference page by means of a new button



This is the most convenient way of building from command-line with the same settings as in the UI.

- It is also possible to specify path maps in a text file



Model Compiler Improvements (4/4)

- More flexible ways to specify source locations when building from command-line
 - Multiple `--root` arguments can now be used
 - Possible to use a workspace with the `--root` argument
 - Possible to search in subfolders
- Support for environment variables for the `--cwd` option



Model Compiler Preferences

- A new preference allows you to set the Java Virtual Machine arguments to use when launching the model compiler
 - Available on the preference page *UML Development – Real Time C++ Transformations*
 - For example, you can use it to increase the memory for the model compiler if needed

VM arguments:

- A new preference allows you to specify the port range to use by the model compiler server (the first available port in the range will be used)
 - Available on the preference page *UML Development*
 - There is also a button for restarting the server (you don't normally have to do this)

Model Compiler Server Settings

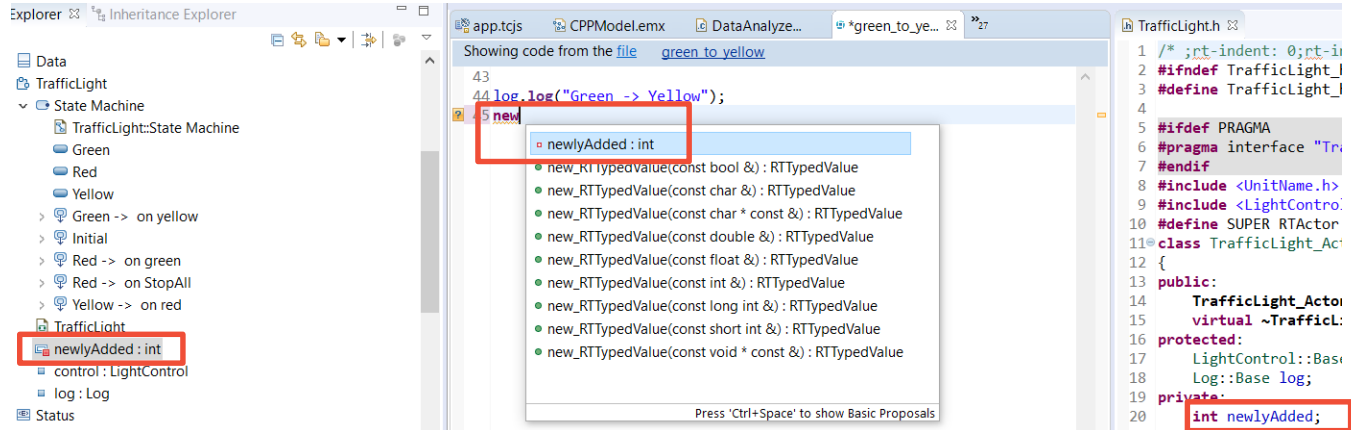
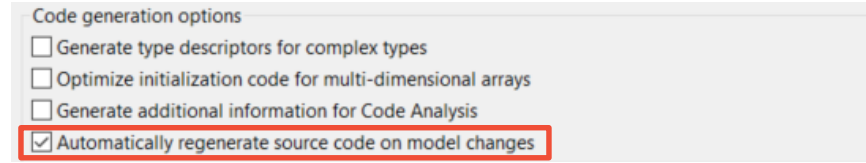
Port range (syntax is low:upper), restart is required :

Model Compiler Server status: server is running at port 60004



Automatic Code Generation

- A new preference allows you to automatically generate code
 - Code generation is triggered when a modified model file is saved
 - Makes it easier to use CDT Content Assist (no longer needed to first build the TC)
 - By docking a CDT editor window you can get an immediate preview of generated code while editing
 - Note: Save a modified code snippet (to trigger code generation) before invoking Content Assist.
 - This feature is only available when using the Model Compiler.



TargetRTS

- The RSARTE installation now contains a pre-built TargetRTS library for new MinGW
 - It has been built with MinGW 8.1.0 64 bit
 - The old version of the MinGW TargetRTS is still included but it's recommended to switch to the new TargetRTS if possible

TargetRTS configuration:

MinGwT.x64-MinGw-gnu-8.1.0

Make type:

MinGwT.x64-MinGw-gnu-8.1.0

Compile arguments:

NT40MinGwT.x86-MinGw-gnu-4.7.0

NT40T.x64-VisualC++-9.0

NT40T.x86-VisualC++-9.0

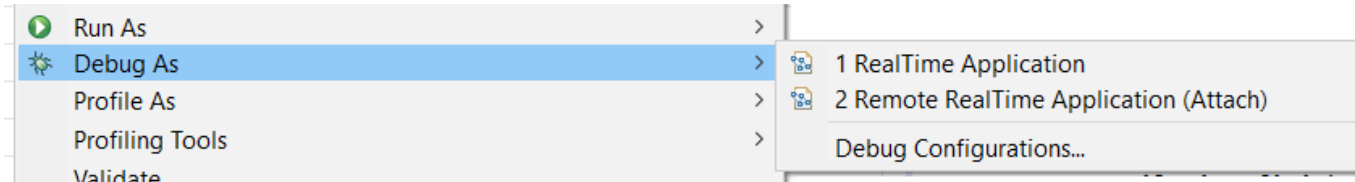
Compile command:

SLED10x64T.x64-gnu-4.1

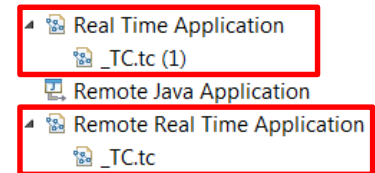


Model Debugger (1/4)

- The "model execution" feature has evolved into a Model Debugger feature
 - Everything in the user interface related to general "model execution" was removed since it's not applicable to RSARTE. For example, the Model Execution perspective is now called Model Debug and the Model Execution preference page was simplified to *Run/Debug – RealTime Application*.
- New commands are available in the context menu of an executable TC for launching a model debug session
 - It is possible to either launch a model debug session locally, or to attach the model debugger to an already running application (possibly running on a different machine)

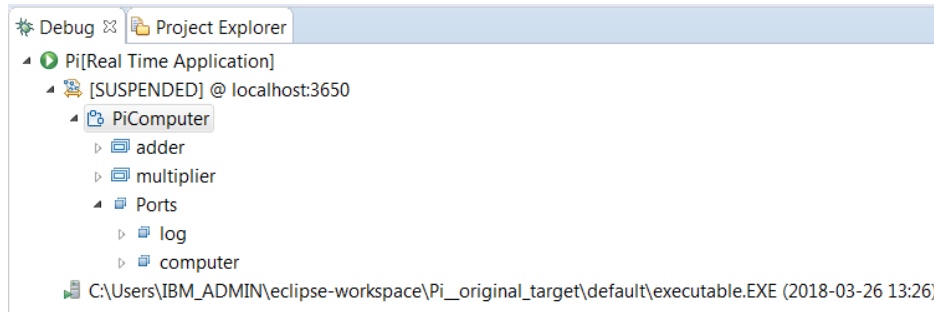


- These commands use new launch configurations (created automatically as needed)
- The commands are also available for TCs in the new .tcjs format

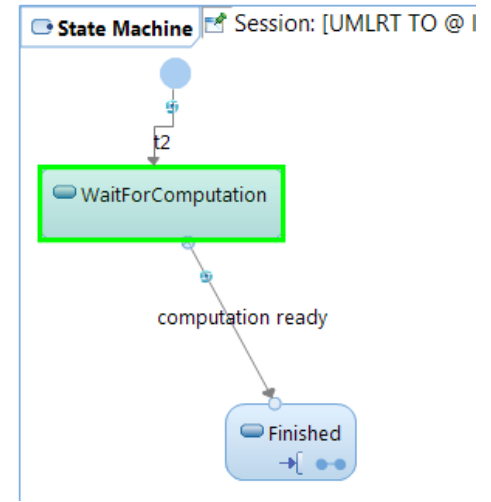


Model Debugger (2/4)

- The model debugger allows you to inspect what the real-time application is doing and to control its execution
- The current structure and state of the application can be inspected using the Debug and Variables views and in instance diagrams

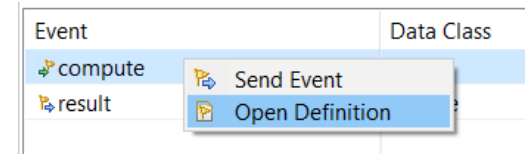
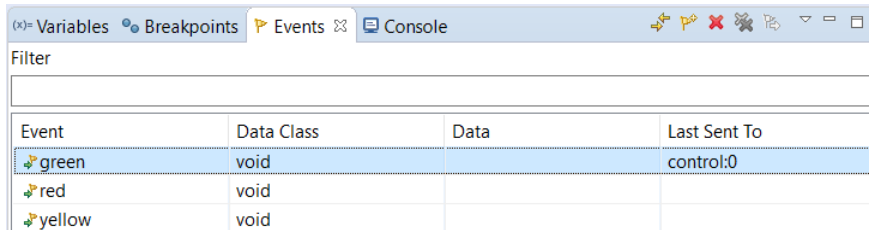


- Breakpoints can be used for suspending execution at interesting places (states, transitions and ports)
 - Commands for setting code-level breakpoints have been removed (use CDT commands instead for setting such breakpoints)

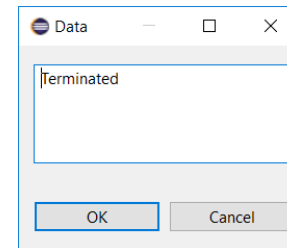
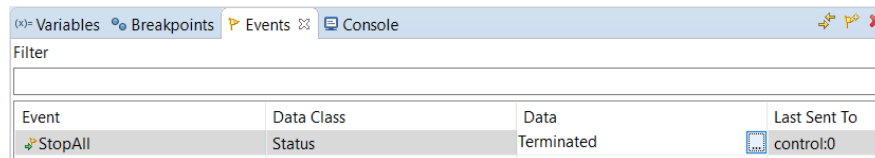


Model Debugger (3/4)

- The Events view allows you to send events to the application through ports
 - Send events by drag/dropping them onto ports in the Debug view
 - You can navigate to the corresponding protocol event from the context menu
 - It's possible to populate the Events view by drag-and-drop of protocol events from the Project Explorer
 - Specify data for an event using the Data column
 - The "Last Sent To" column is set when an event is sent and enables future sending of that event using double-click

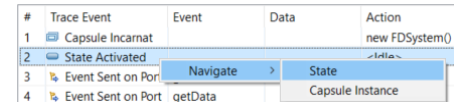
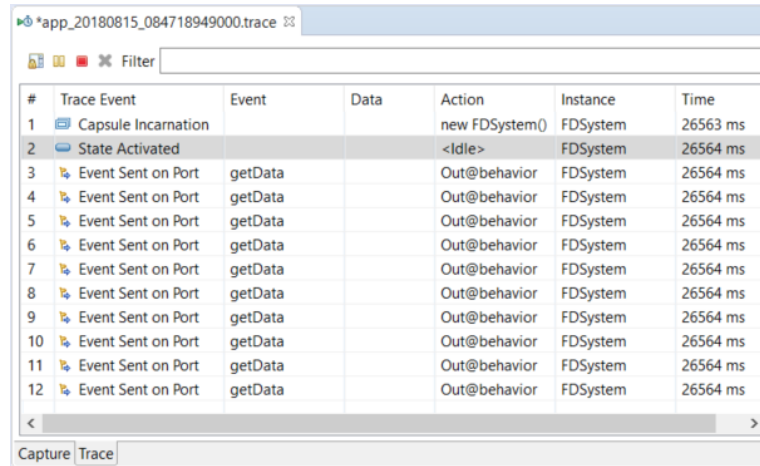
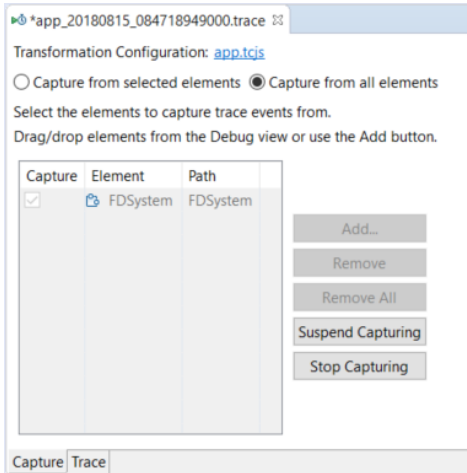


- Event data can be edited using a dialog



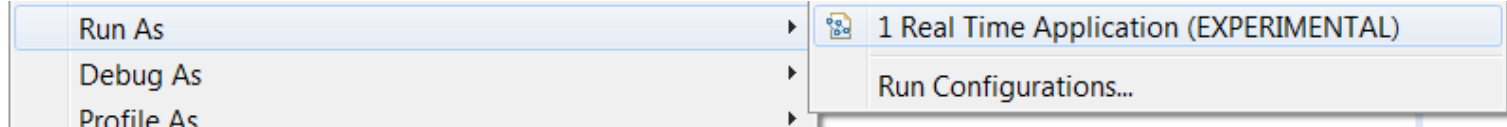
Model Debugger (4/4)

- A Trace editor can be used for tracing what the application is doing
 - Replaces the old Trace view which now has been removed
 - Multiple elements can be traced into the same trace file (textual file visualized in trace table)
 - Trace table can be filtered and individual trace events search for (incremental and regular search)
 - Possible to navigate from trace events to the model

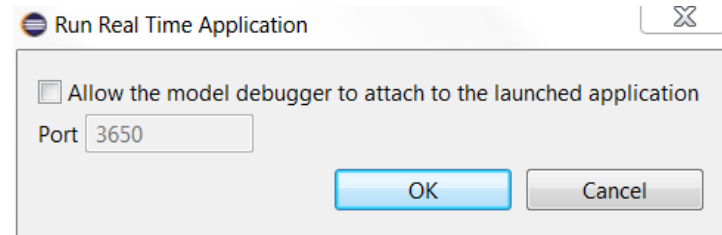


Launching Generated C++ Applications

- The special launch configuration "UML Capsule C/C++ Application" has been removed
- It was replaced by a new command in the context menu of an executable TC for launching the application built by that TC

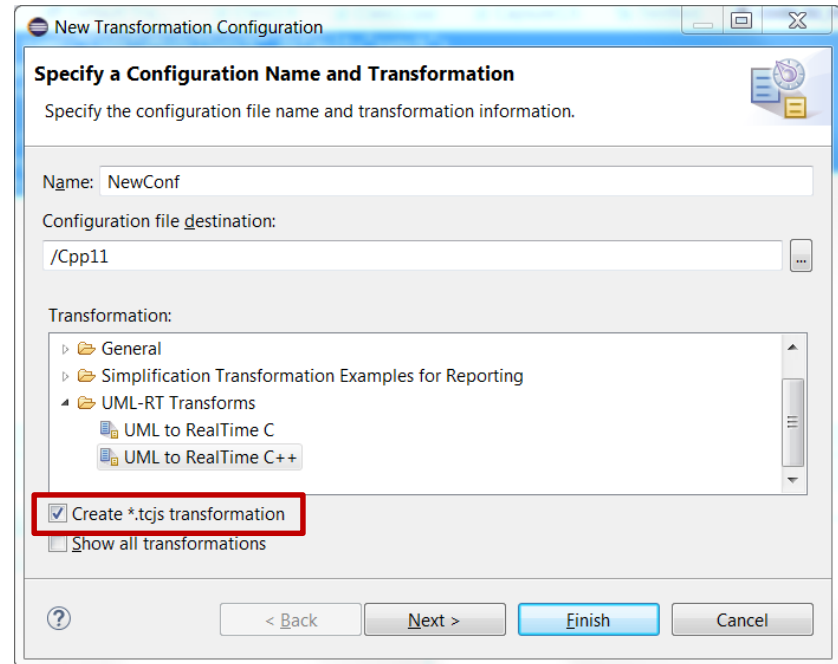


- This command (still experimental) uses a regular C/C++ Application launch configuration with command-line arguments appropriate for running the application without debugging. The launch configuration is created automatically (if it does not already exist). If needed you can edit it manually in the launch configuration dialog.
- A dialog asks if the application should be launched in a way so that you later can attach the Model Debugger to it.



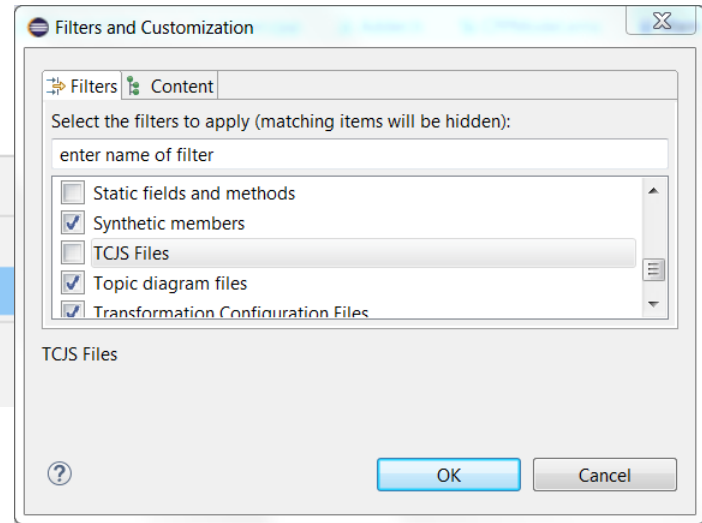
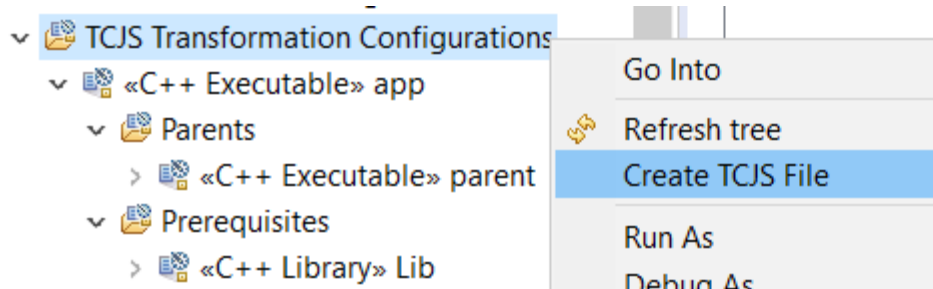
New File Format for Transformation Configurations

- The file extension *.tcjs is now used for TC files in the new JavaScript format
 - Eclipse had some limitations when using the previous "double extension" format (*.tc.js)
- The wizard for creating new TCs now supports the new file format
 - Only for "UML to RealTime C++" TCs



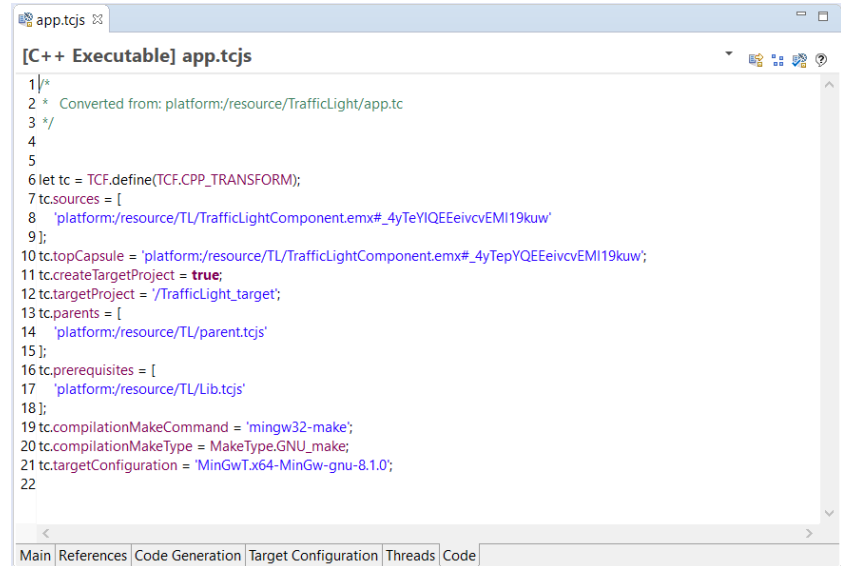
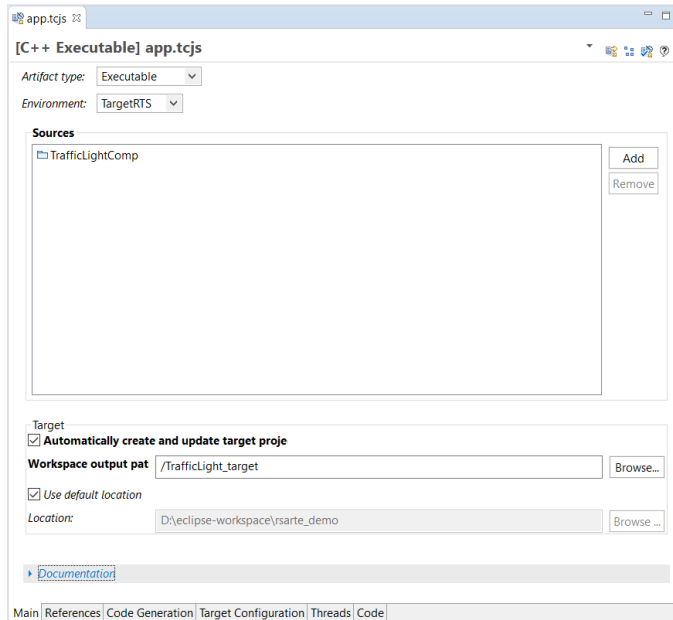
New folder for *.tcjs Files in the Project Explorer

- A virtual folder "TCJS Transformation Configurations" is available in the Project Explorer
 - Similar to the "Transformation Configurations" folder for TCs in the old format, but shows the hierarchy of both parent and prerequisite TCs
 - A Project Explorer filter is available for hiding this folder
 - Use the command *Refresh tree* in the context menu in case you change the .tcjs files outside of RSARTE
 - A context menu command Create TCJS File on this folder allows for quick creation of a new .tcjs file



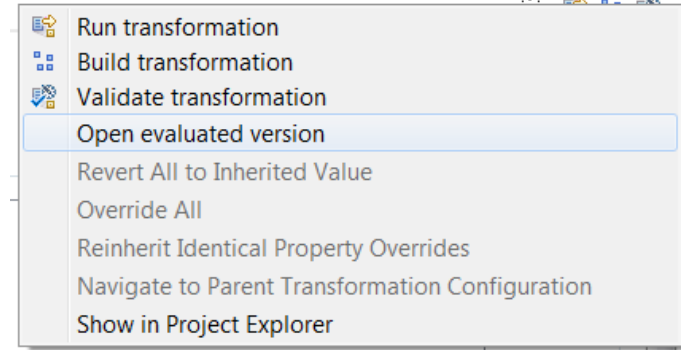
New Transformation Configuration Editor (1/3)

- A new graphical editor is now available for editing TCs in the new *.tcjs format
 - It is similar to the old editor, but properties are organized in a more natural way
 - Also, a new Code tab allows you to edit properties directly in JavaScript syntax

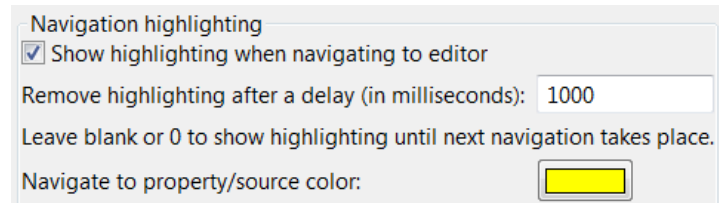


New Transformation Configuration Editor (2/3)

- New command for evaluating TC properties
 - Opens a read-only TC editor that shows the values of all TC properties as they will be when building the TC
 - Useful for troubleshooting build problems caused by TC property values
- Navigation from TC properties in Code tab
 - TC properties in the Code tab can be ctrl+clicked to navigate to the corresponding TC property in one of the other tabs
 - Also, a tooltip now shows the more user-friendly name of the TC property
- New preference to control navigation to TC properties
 - Available in *UML Development – TCJS Editor – Navigation highlighting*
 - You can control the highlight color and the duration when it is shown



```
10 tc.createTargetProject = true;  
11 tc.Create target project 'NT40'  
12 tc.targetProject = /BuildVarian
```



New Transformation Configuration Editor (3/3)

- Context sensitive help is available
 - Press the Help button in the upper right corner of the TC editor
 - Documents all TC properties shown in the current tab

The screenshot displays the Transformation Configuration Editor (TC) interface. The main window is titled "[C++ Executable] app.tcjs" and shows configuration options for a transformation configuration of type "C++ Executable". The configuration is for a capsule named "TrafficLightComp::TrafficLight".

Configuration options include:

- Compile data classes individually
- Enable support for file artifacts
- Generate code qualifiers
- Optimize handling of frequent triggers
- Context sensitive library build

Input fields for:

- Default arguments: []
- Output subdirectory: []
- Unit name: [UnitName]
- Unit subdirectory: []

Additional options:

- Include unit header file without unit subdirectory path
- Comment style: [Default]
- Common preface: []

Buttons: Select... Clear

The Help panel is open, showing the "Properties" section. A tooltip above the Help button reads: "Show help for the current transformation configuration editor tab".

Properties

This page of the transformation configuration editor contains the settings for how to transform the model into C++ code. It also contains settings that control how to generate the makefile to be used for building generated C++ code.

Top capsule This setting is only available for a transformation configuration of type "C++ Executable". It specifies which capsule that should be automatically incarnated when the executable is run. The top capsule is hence the entry point of the application. The top capsule must always be a source element of the transformation configuration, either explicitly or implicitly.

Compile data classes individually By default each data class (a.k.a. passive class) is compiled by itself. Unset this setting to instead compile them as part of the unit .cpp file. In some cases this may lead to faster compilation times.

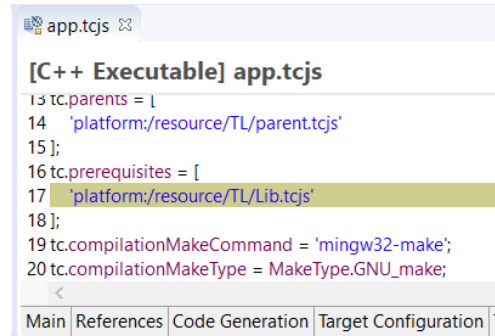
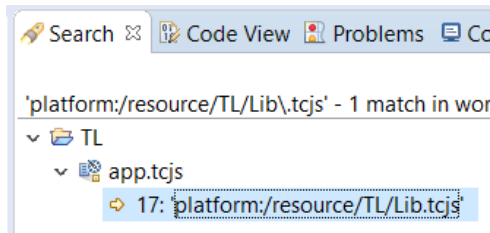
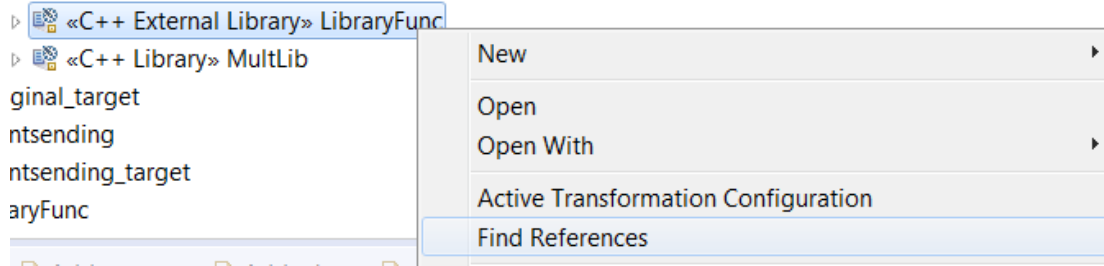
Enable support for file artifacts Enable this setting if your model contains artifact model elements with C++ code snippets. Such artifacts are called file artifacts, and will be translated into C++ header and implementation files. A file

Navigation tabs at the bottom: Main | References | Code Generation | Target Configuration | Threads | Code



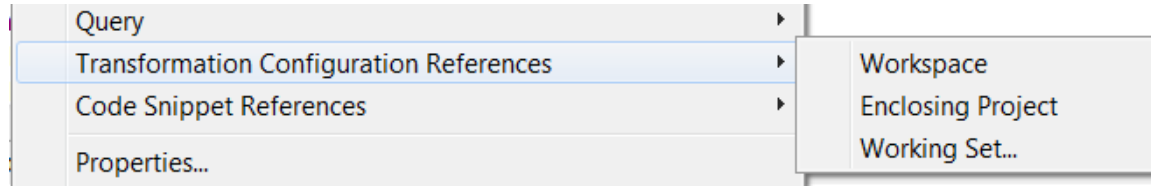
Finding References in TCs (1/2)

- To find references to a TC in the new format, use the context menu command *Find References*
 - Finds TCs containing references to the selected TC and shows them in the Search view
 - Navigating a found reference opens the TC editor and highlights the reference in the Code tab



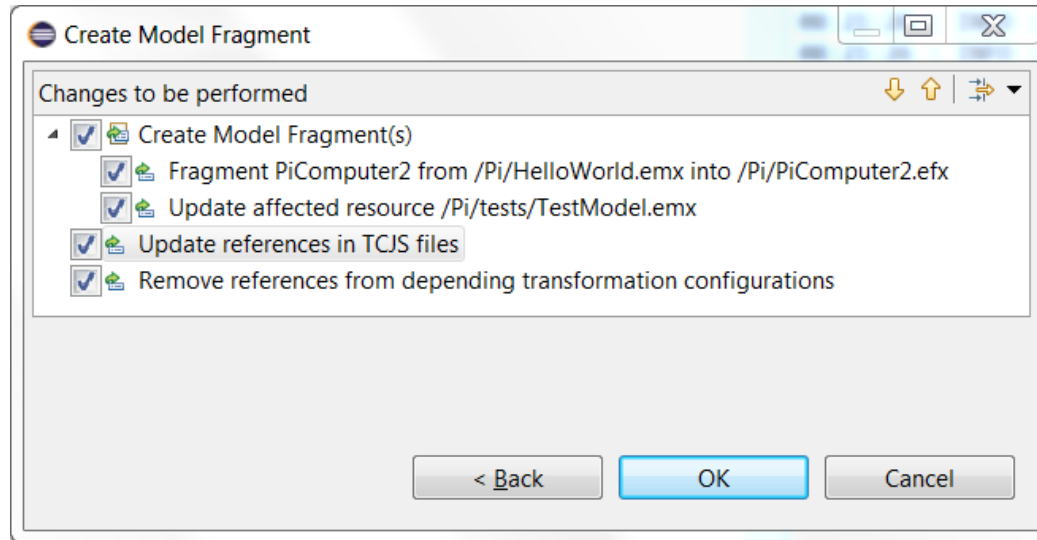
Finding References in TCs (2/2)

- The command for finding references to model elements in TCs now work also for TCs in the new format
 - Use the commands in the *Transformation Configuration References* context menu



Updating TCs when Refactoring

- TCs in the new format are now correctly updated when you perform a refactoring
 - For example, when creating a new fragment file for the top capsule, references to it in TCs will also be updated



Build Variant and Transformation Configuration APIs

- The APIs have now been documented in online help
 - Link available in the API Reference of "RSARTE Transformation Developer's Guide"

RSARTE Transformation Developer's Guide

Reference

API Reference

Extension Points Reference

UML2 Developer Guide

- [Mapping Transformation Utility](#)

RSARTE also contains a JavaScript framework that allows you to work with transformation configurations. You can use this API to read and write values of transformation configuration settings, for transformation configurations defined in the new JavaScript based file format (*.tc.js). You can also use this API to define build variants. A build variant script defines possible ways to customize the build of a transformation configuration. It also allows you to define a user interface for choosing which variant of an application to build.

- [Transformation Configuration and Build Variants JavaScript API](#)

© Copyright IBM, HCL and others 2004, 2018

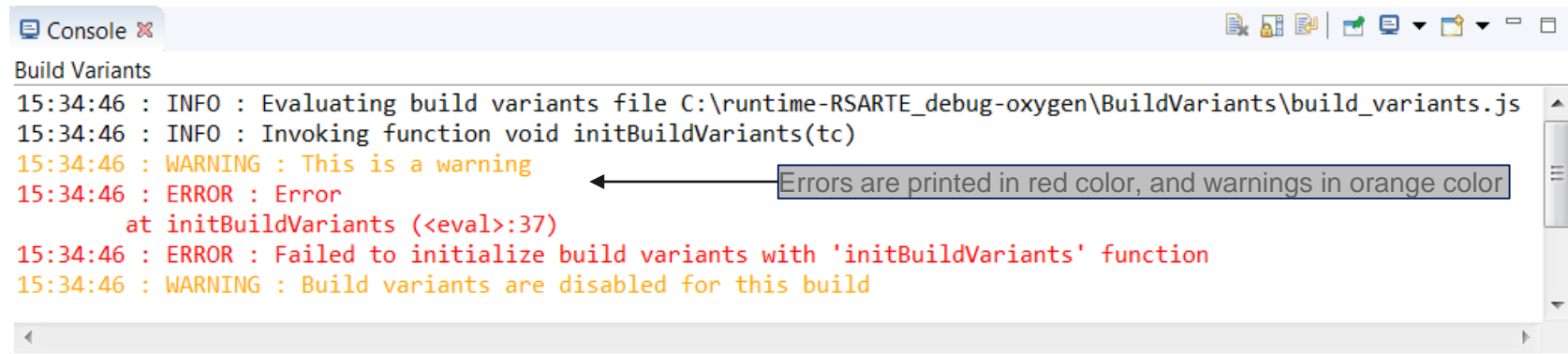
- The Build Variant feature is now non-experimental



Build Variants Console

- A new Build Variants console is now available
 - Messages printed with `BVF.formatInfo()`, `BVF.formatWarning()` and `BVF.formatError()` are printed there
 - This significantly helps "debugging" problems when writing build variant scripts

```
function initBuildVariants(tc) {  
    BVF.add(debug, target);  
    BVF.formatWarning('This is a warning');  
    BVF.formatError(new Error().stack);  
}
```

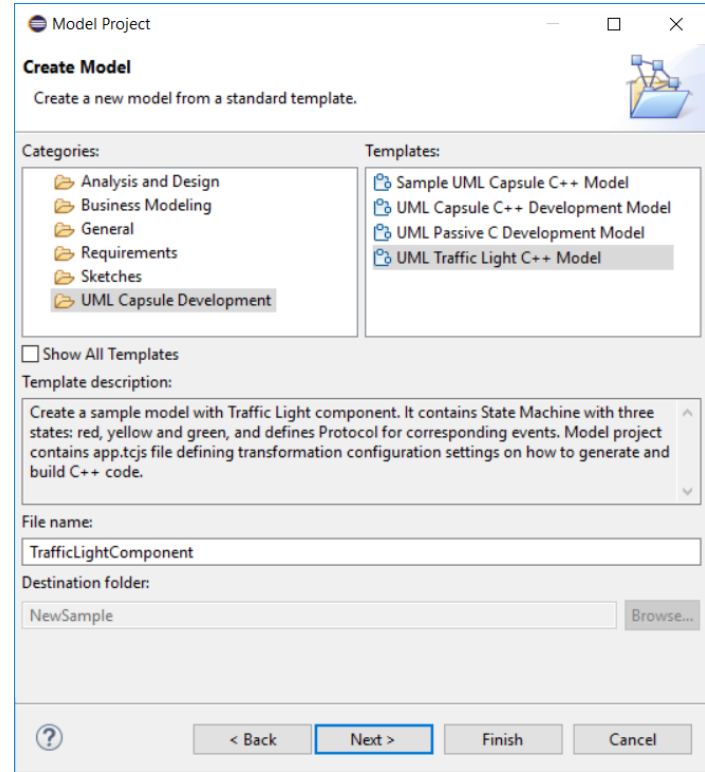
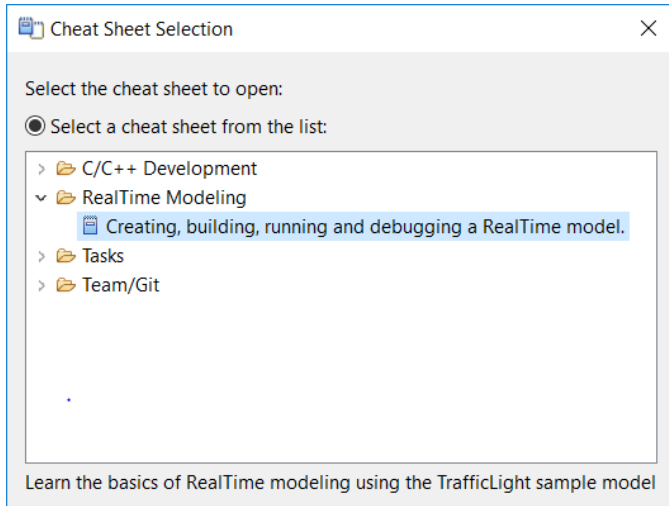


```
Console x  
Build Variants  
15:34:46 : INFO : Evaluating build variants file C:\runtime-RSARTE_debug-oxygen\BuildVariants\build_variants.js  
15:34:46 : INFO : Invoking function void initBuildVariants(tc)  
15:34:46 : WARNING : This is a warning  
15:34:46 : ERROR : Error  
                at initBuildVariants (<eval>:37)  
15:34:46 : ERROR : Failed to initialize build variants with 'initBuildVariants' function  
15:34:46 : WARNING : Build variants are disabled for this build
```



New Sample Model and Cheat Sheet

- A new sample TrafficLight is now included in RSARTE
- A new cheat sheet is also available
 - Introduces common RSARTE scenarios using the TrafficLight sample



Documentation Improvements

- Articles from the RSARTE DeveloperWorks wiki are now included in the built-in Help
 - Located in the RSARTE User's Guide under **Wiki**
- Almost all PDFs are now included in the built-in Help
 - Previously these PDFs were only available on DeveloperWorks, but are now also included in the built-in Help
- A new document about TargetRTS changes is now available
 - Available on the help page "The RT Services Library"
 - Contains all changes in the RSARTE TargetRTS compared to the version used in Rose RT



[All Changes in C++ Target RTS from Rose RT to RSARTE.pdf](#)

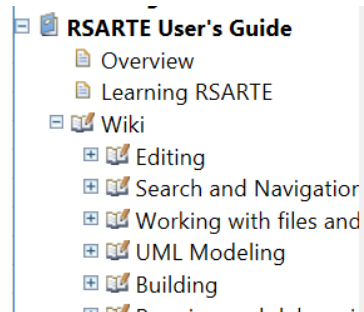
Describes all changes in C++ Target RTS service library implemented between the latest version of Rational Rose RT and current version of RSARTE.

- A new document about the model debugger is now available



[Model Debugger](#)

Describes how to debug applications generated by RSARTE using the model debugger.



THANK YOU!