What’s New in HCL RTist 11.1
updated for release 2021.34
Overview

- RTist 11.1 is based on Eclipse 2020.06 (4.16)

- HCL RTist is 100% compatible with IBM RSARTE. All features in IBM RSARTE are also present in HCL RTist. However, HCL RTist contains some features that do not exist in IBM RSARTE.
  - Those features are marked in this presentation by **RTist only**

(c) Copyright IBM Corporation 2004, 2016. All rights reserved.
(c) Copyright HCL Technologies Ltd. 2016, 2021. All rights reserved.
Compared to RTist 11.0, RTist 11.1 includes new features from 4 quarterly Eclipse releases:


For full information about all improvements and changes in these Eclipse releases see the links above

- Some highlights are listed in the next few slides...
A new Quick Search dialog allows you to search the files of your workspace faster (“as-you-type”)

- For a similar search experience in model files, use the Find Named Element command instead

![Quick Search Dialog](image)
Eclipse 4.16 (2020.06)

- By default at most 99 editors can now be open at the same time
  - Helps keeping the performance good when working with Eclipse for a long time
  - This can be controlled by the preference **General – Editors – Close editors automatically**

- Showing key bindings when performing commands
  - New preferences in **General – Keys**
  - This is a good way to learn about key bindings for the commands that are used, and can also help in presentations

**Ctrl+T – Open State Machine Diagram**
Quick Access field replaced with toolbar button

- Takes less space in the toolbar, and instead uses a normal dialog for typing and showing the results
- Same key binding as before (Ctrl + 3) but the command is now called “Find Actions”
- The results now also include matching files in the workspace, and text matches in files (requires that Quick Search has been used at least once)
Eclipse 4.16 (2020.06)

- Show code problems inline
  - Makes errors/warnings more visible and lets you apply quick fixes without having to go to the Problems view
  - Enable this feature in preferences at General – Editors – Text Editors – Show code minings for problem annotations

- There were several improvements in SWT and GTK
  - The minimal supported GTK version is now 3.20
New Debug Sources view

- Shows source files the C++ debugger knows about when debugging an application
- Useful in particular when the application contains source files that are not present in the Eclipse workspace
- Source files can be found by searching (filtering) and opened by double-click
CDT 9.11 (included as part of Eclipse 2020.06)

- CODAN improvements
  - Several additional checks implemented

- For more information about CDT improvements see
  https://wiki.eclipse.org/CDT/User/NewIn910
  https://wiki.eclipse.org/CDT/User/NewIn911
Newer EGit Version in the EGit Integration

- The EGit integration in RTist has upgraded EGit from 5.4 to 5.8
  - This is the recommended and latest version for Eclipse 2020.06

- This upgrade provides several new features, performance improvements and bug fixes
  - For detailed information about the changes see
    https://wiki.eclipse.org/EGit/New_and_Noteeworthy/5.5
    https://wiki.eclipse.org/EGit/New_and_Noteeworthy/5.6
    https://wiki.eclipse.org/EGit/New_and_Noteeworthy/5.7
    https://wiki.eclipse.org/EGit/New_and_Noteeworthy/5.8
A bash script is now available which helps automating the installation of RTist

- Download it from the [Info Center](#)
- Works on both Windows and Linux

In particular useful for installing RTist 11.1 (due to the requirement of using Java 11 for the installation)

- Choose whether you want to then run RTist with either Java 8 or Java 11

For documentation on how to configure and use the script see the [Info Center](#).
The Default Value field now supports multi-line values

- To create a multi-line default value you still need to use the Code View or Code Editor
- For editing a multi-line default value you can now use the Properties view, but it’s still often more convenient with the Code View or Code Editor
- For quickly viewing a multi-line default value the Properties view can be handy
The Project Explorer can now show template information after the name of an element that has template parameters

- Makes it easier to see if an element is a template without having to expand it in the Project Explorer, or look in the Properties view

- A new preference **RealTime Development – Project Explorer – Show Template Parameters in Labels** controls what to show

![Screenshot of settings](image-url)

- **Do not show template parameters**
- **Show existence of template parameters**
- **Show full template parameter list**
Search Filtering

- It’s now possible to filter search results using Boolean operators NOT (!) and AND (&&)
  - Useful if a search returns too many matches
  - Use a filter on the form !A && !B && ... !X to hide matches where certain words are not present
  - Use a filter on the form A && B && ... X to only show matches where certain words are present
  - ...or any combination, where some words are present and others not
- Enclose the filter string in double quotes to apply the filter verbatimly
  - Needed if the filter string contains the characters ! or &&
 Enums with Operations

- Enumerations can now have operations
  - Create them as usual with **Add UML - Operation**

- Such operations will be translated to global functions
  - C++ enums cannot have member functions, but it’s sometimes useful to have functions that operate on or return enum literals
  - Using global functions can then be an alternative to wrapping the enum inside a class

- This works the same both for scoped and non-scoped enumerations
Generic Type Descriptors

- The model compiler now supports generating type descriptors for type aliases with template parameters
  - For example: `template<typename T, unsigned int N > using StdArray = std::array<T, N>;`
  - If type descriptor functions are defined for the type alias, they will be generated as template functions with the same template parameters
  - Allows to implement generic type descriptors that work for all (or many) instantiations of the template
  - A new `RTObject_class::fromType<T>()` template function can be used for looking up the type descriptor of a type at compile time. Useful for example when implementing generic encode or decode functions. Specialize it for the types that you use (specializations for built-in types are available in the TargetRTS). For example:
    ```cpp
    template <> inline const RTObject_class* RTObject_class::fromType<RTString>() {
      return &RTType_RTString;
    }
    ```
- You can specify a unique name for the type descriptor of a specific template instantiation
  - For example: `template <> const char* RTName_StdArray<StdString, 4>::name = "StdArray<StdString, 4>";`
  - The TargetRTS now prints a warning if two type descriptors with the same name exists. Helps troubleshooting missing template specializations for the name attribute.
A new group of preferences were introduced to let the model compiler generate code according to certain code compliance rules

- As a first step support for one specific Clang-Tidy rule is implemented
- It suppresses warnings for use of static_cast to downcast event data in transition functions

```cpp
transition2_t1( static_cast< const bool * > ( msg->data ), static_cast< P::Base * > ( msg->sap() /* NOLINT(cppcoreguidelines-pro-type-static-cast-downcast) */ ) );
```
Error Message when Failing to Delete Files or Folders

- Certain commands in RTist involve deletion of files and/or folders
  - Cleaning a TC
  - Removing code preview
  - ...etc

- Now, if the required files or folders cannot be deleted, a clear error message is shown
  - Previously there would be a silent failure in such situations which could be hard to understand the reason for
  - The new message is identical to what Eclipse would show if you directly try to remove the files/folders from the Project Explorer. Click the Details button to see exactly which file or folder that couldn’t be deleted, and why.
Support for Path Variables in Transformation Configurations

- Path variables can now be used in certain TC properties
  - Useful for those TC properties that specify a path
  - Define path variables in Preferences at **General – Workspace – Linked Resources**
  - This can be an alternative to using string substitutions (**Run/Debug – String Substitutions**) or environment variables in order to have a more generic TC (a path variable takes precedence over other kinds of variables, if the same variable name is used).

- The model compiler now prints a warning if a variable used in a TC property cannot be resolved

```
WARNING: Cannot resolve variable '$(TARGET_DIR)' in 'Location' property: '$(TARGET_DIR)'
```
Mocha is a popular JavaScript framework for testing asynchronous applications

It’s now possible to use Mocha also for unit testing capsules

- Provided by a new component that can be selected when installing
- Note that it depends on NodePlus and is currently an EXPERIMENTAL feature

To create a Mocha unit test for a capsule, invoke the new context menu command **Add Unit Test**
The **Add Unit Test** command creates everything necessary for writing a unit test for the capsule:

- A test driver model where all service ports of the capsule under test ("cut") are connected to similar but conjugated ports of a test probe capsule.
- A TC for building the test driver model into an executable that uses the TcpServer library for exposing all test probe ports to the Mocha test script.
- A Node.js project with a Mocha test script ready to implement the unit test.
The unit test can be executed right away

- Build the test driver TC (only needed the first time, and whenever you change the capsule under test)

- Install the Node.js dependencies for the JavaScript project (right-click on the project and do Run As – npm install (only needed the first time – it is assume you already have installed Mocha on the machine)

- Run the testcase by right-click on the .js file and do Run As – JavaScript Unit Test

The test execution result is shown in the JavaScript Unit Test view
Reporting with BIRT

- Create reports that include information from an RTist model
  - Same capabilities as in RTist 10.3, but now adapted for recent Eclipse versions (supports RTist 11.0 and RTist 11.1)
  - Delivered as a separate update site on our InfoCenter. Installation instructions are included in the ZIP file.
  - This is currently an experimental feature