Side by Side Comparison
RSARTE versus Rose RT

The main reasons why you should migrate

Eclipse



- RSARTE is installed on latest Eclipse (4.7 Oxygen)
  - Open source IDE with very big and active user community
  - Lots of useful Eclipse plugins are available for tailoring the IDE for your needs
  - Endless possibilities of customizations (by writing Java plugins)
  - Works equally well on Windows and Linux platforms with native look-and-feel
  - Supported on 64 bit platforms
- Rose RT is a proprietary IDE
  - Built on old Windows technology and made available on Linux using porting software. No native look-and-feel on Linux.
  - No active user community means small possibilities of customizing and extending the IDE. Very hard to customize on Linux due to the use of porting software.
  - Only supported on 32 bit platforms

Some Benefits Brought by Eclipse

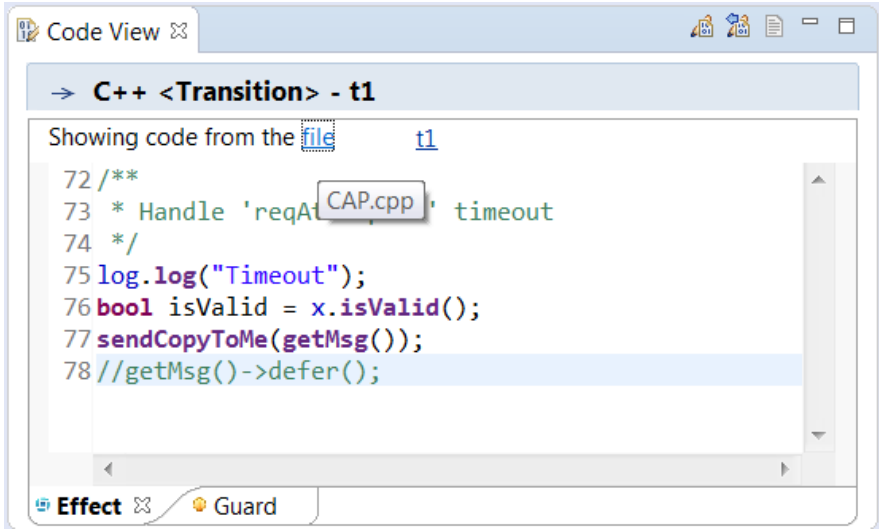
- Multi-project environment
  - Pure C++ projects can be used together with model projects
  - While Rose RT only can support working with one model, it’s possible in RSARTE to split a big model into several smaller model projects
- Perspective and capabilities
  - For adapting the user interface to the role of the user
- Many generally useful IDE features
  - File search, help system, working set filters, console view, problems view etc.
- Keyboard accessibility and shortcuts
- Better management of preferences
- Quick and easy install experience
  - Possible to customize and automate installation using Oomph
- ...and much, much more

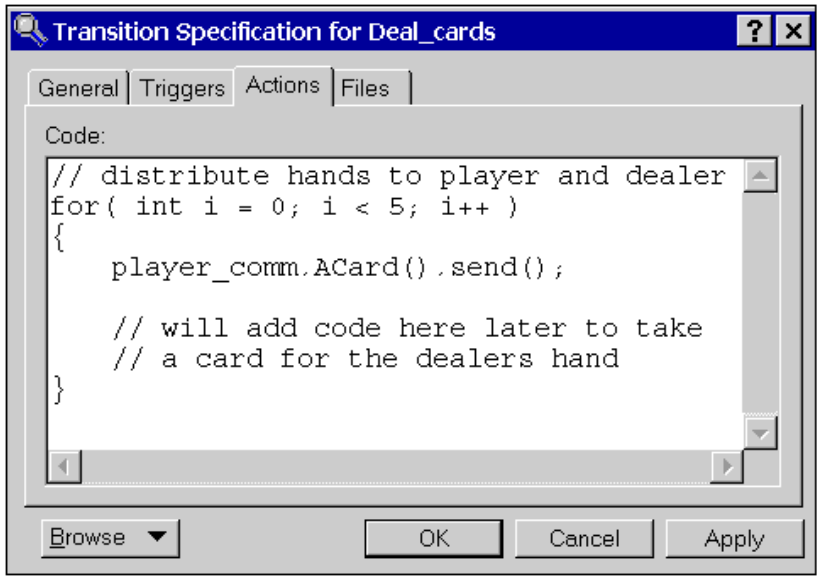
Model Language

- RSARTE is based on UML 2.x
  - Support for a wide range of modeling constructs also outside of the RT domain
  - Better support for end-to-end development workflows involving different roles in SW development (viewpoints are used for adapting the user interface for the needs of different user groups)
  - Support for informal modeling using sketch diagrams
- RoseRT is based on UML 1.x
  - Limited support for modeling constructs outside of the RT domain
  - Mostly intended for modeling for code generation
- Note that RT specific concepts such as capsules, protocols etc. remain the same in RSARTE as in RoseRT

C/C++ Development

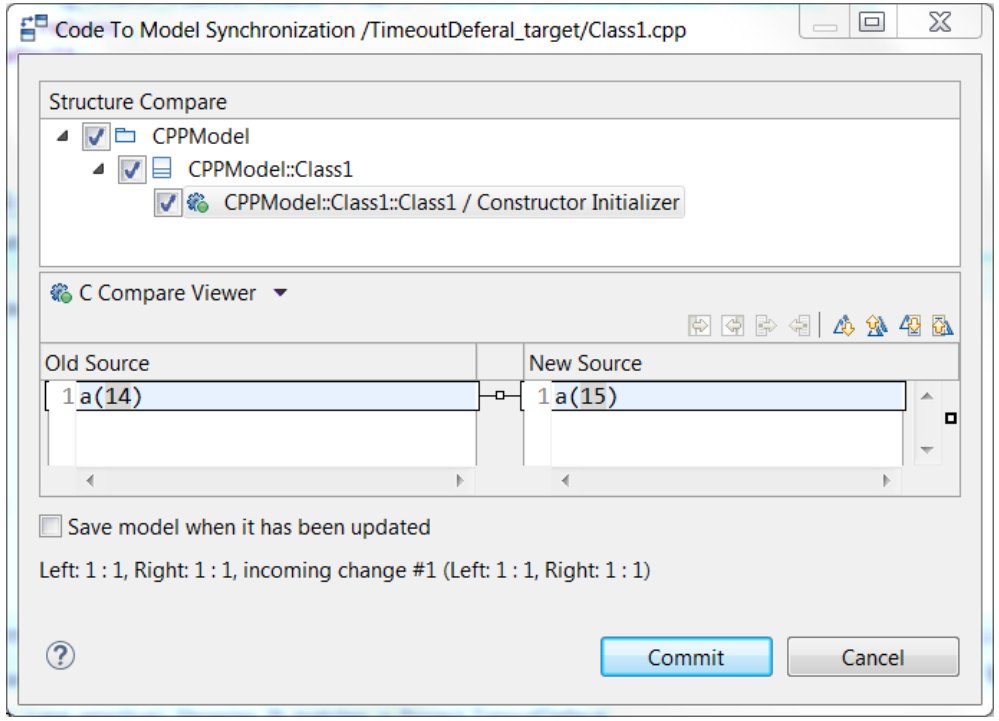
- RSARTE uses the CDT from Eclipse for C/C++ development
  - A Code view and Code editor is available where all code is viewed and edited. It is based on the CDT Editor which means all CDT capabilities are available, such as code completion, syntax coloring, navigation, and much more.
  - CDT projects are generated by the code generator
  - CDT also provides features such as code analysis, refactoring support, debugging etc.
- Rose RT lacks a good C/C++ development environment
  - Code usually has to be edited in dialogs or property views without the modern editing features expected by C/C++ developers today
  - No generation of CDT projects. Need to maintain such projects manually if using Eclipse as development IDE.



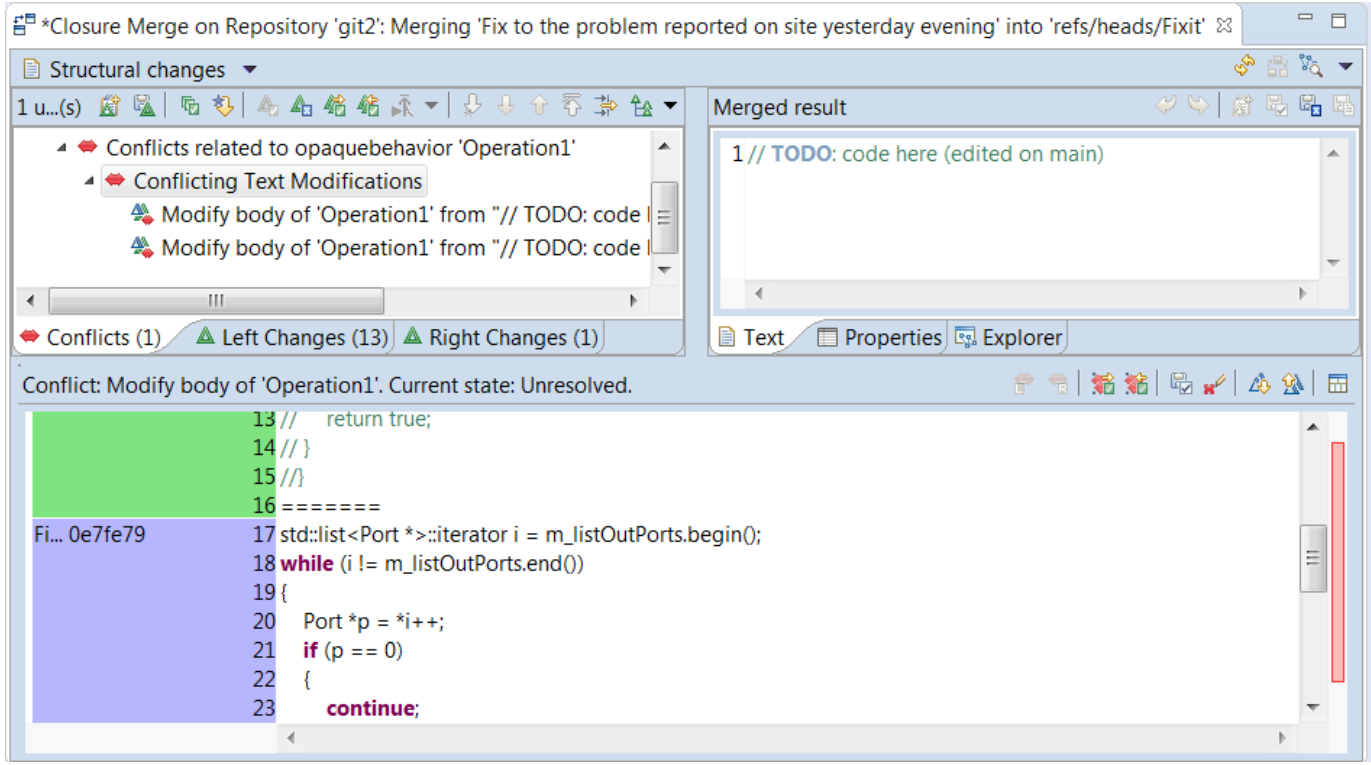


Code to Model Synchronization

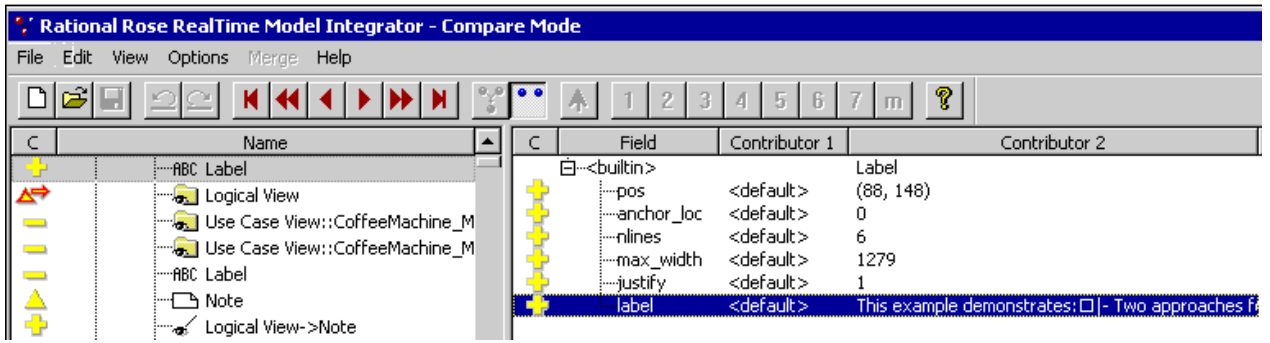
- RSARTE has an improved code-to-model synchronization
  - More kinds of code snippets can be synchronized
  - The changes can be visualized before the model is updated
  - Synchronization of multiple files can be performed as a batch job
- Code Sync in Rose RT has limitations
  - Risk of loosing your code changes in some scenarios



Compare/Merge



- RSARTE provides good support for comparing and merging models
  - Can work on either file, model or closure (multiple models) level
  - Graphical UI at same abstraction level as when constructing the models
  - Intuitive textual merge of code embedded in the model
- Rose RT lacks good support for comparing and merging models
  - The Model Integrator is not a visual model or semantic-level merge tool
  - Complex to use, and difficult to interpret reported changes
  - High risk of corrupting models while merging them



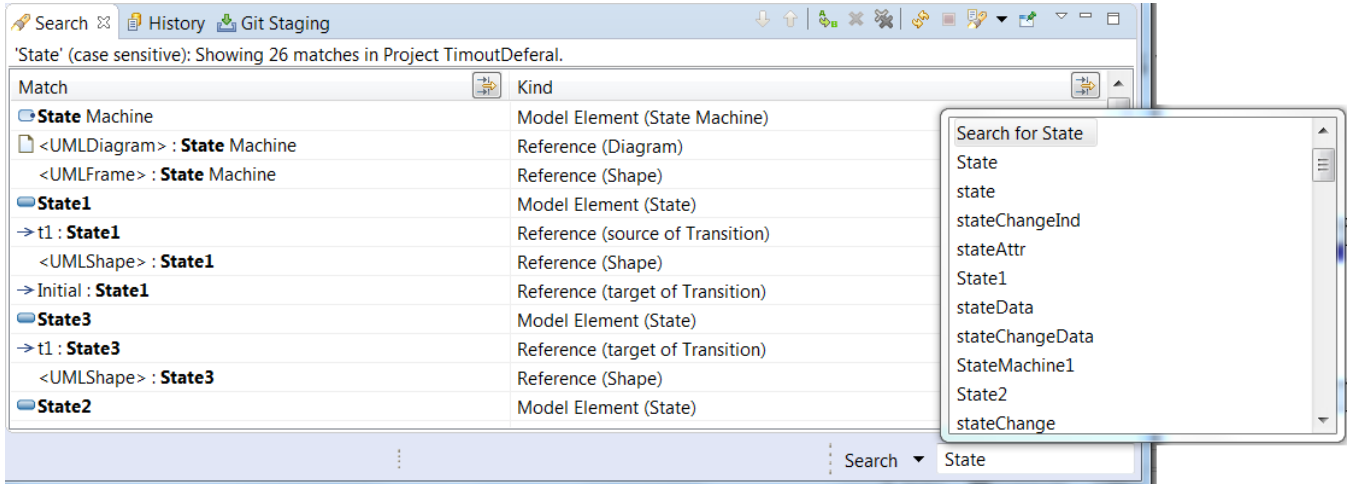
Git Support

- RSARTE provides a powerful integration with Git
  - Built as an extension of the open-source EGit plugin for Eclipse
  - Seemlessly integrated into all graphical views provided by EGit
  - Support for running compare/merge from the command-line with Git specific commands
- Rose RT has no integration with Git
  - The Git command-line has to be used, and it’s necessary to frequently reload files in Rose RT to synchronize them

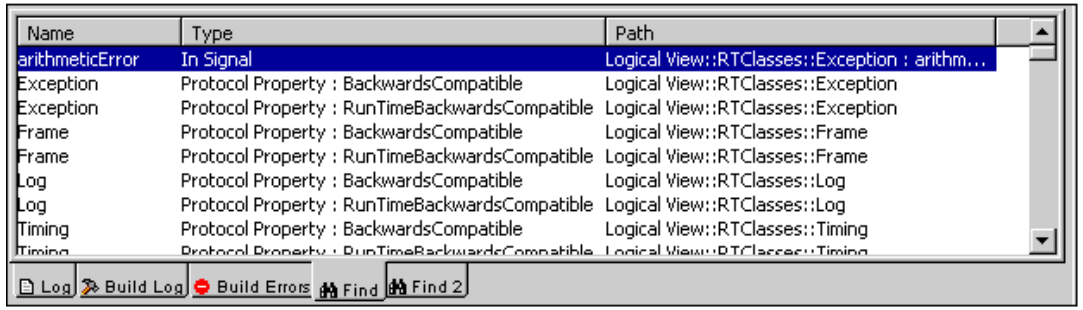
Build Performance and Batch Builds

- RSARTE provides an improved tool for building models (”model compiler”)
  - Allows to build models very fast from the command-line (batch builds) without any dependency to the Eclipse user interface
  - Code generation can be performed from a make file, so that make can drive the full build process. This can allow for parallel code generation and compilation which increases build performance.
  - Can also be used when building models from the user interface (easy single click solution). The build runs in the background and the user can continue to work in the model.
- The Rose RT code generator is less sophisticated
  - Generates makes files and source code first, and then invokes make. Not possible to drive the full build process from a single make file.

Search and Navigation

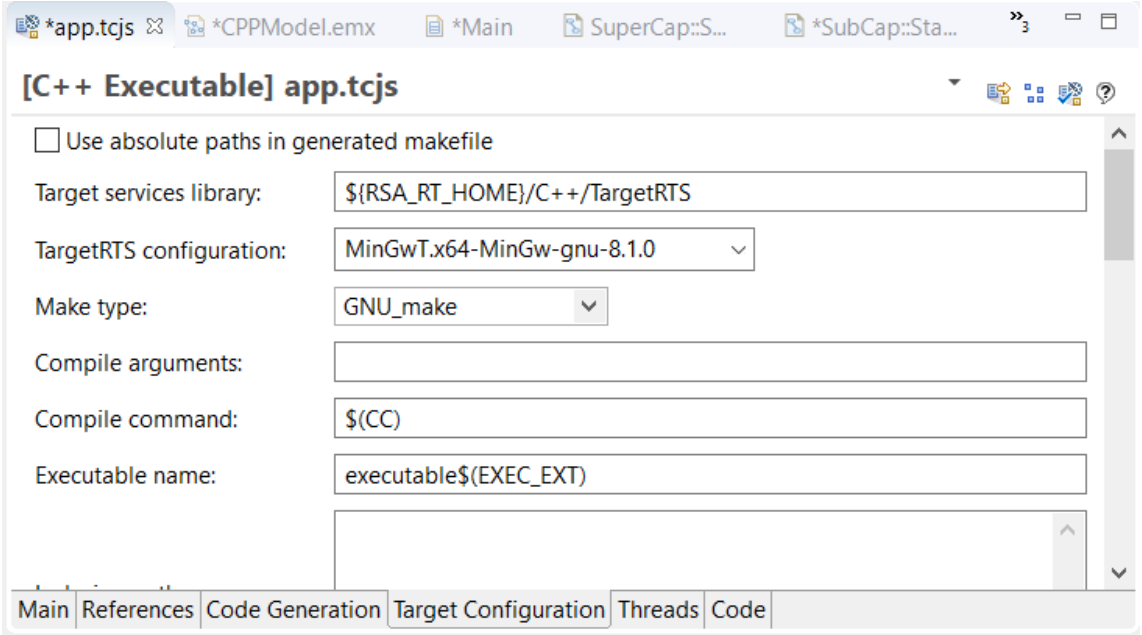


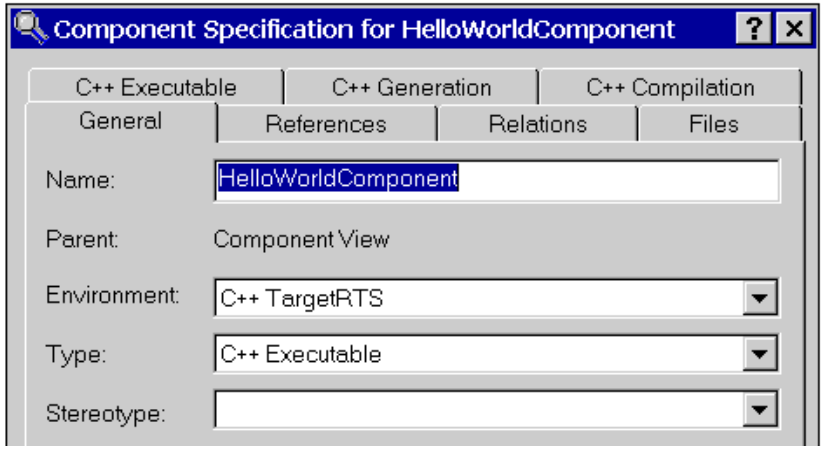
- RSARTE provides many navigation commands and a powerful search
  - A ”Google-style” search field with autocompletion supports searching both in model, code, diagrams and in transformation configurations (build settings)
  - Possible to search in models located outside of the workspace
  - Search result can be filtered using regular expressions
  - Many navigation commands in diagrams, Project Explorer, Properties view etc.
  - Incremental textual search in diagrams
  - Bookmarks and navigation history (to quickly return to interesting model elements)
- Rose RT search and navigation capabilities are limited
  - Only 2 search results can be shown at the same time



Transformation Configurations

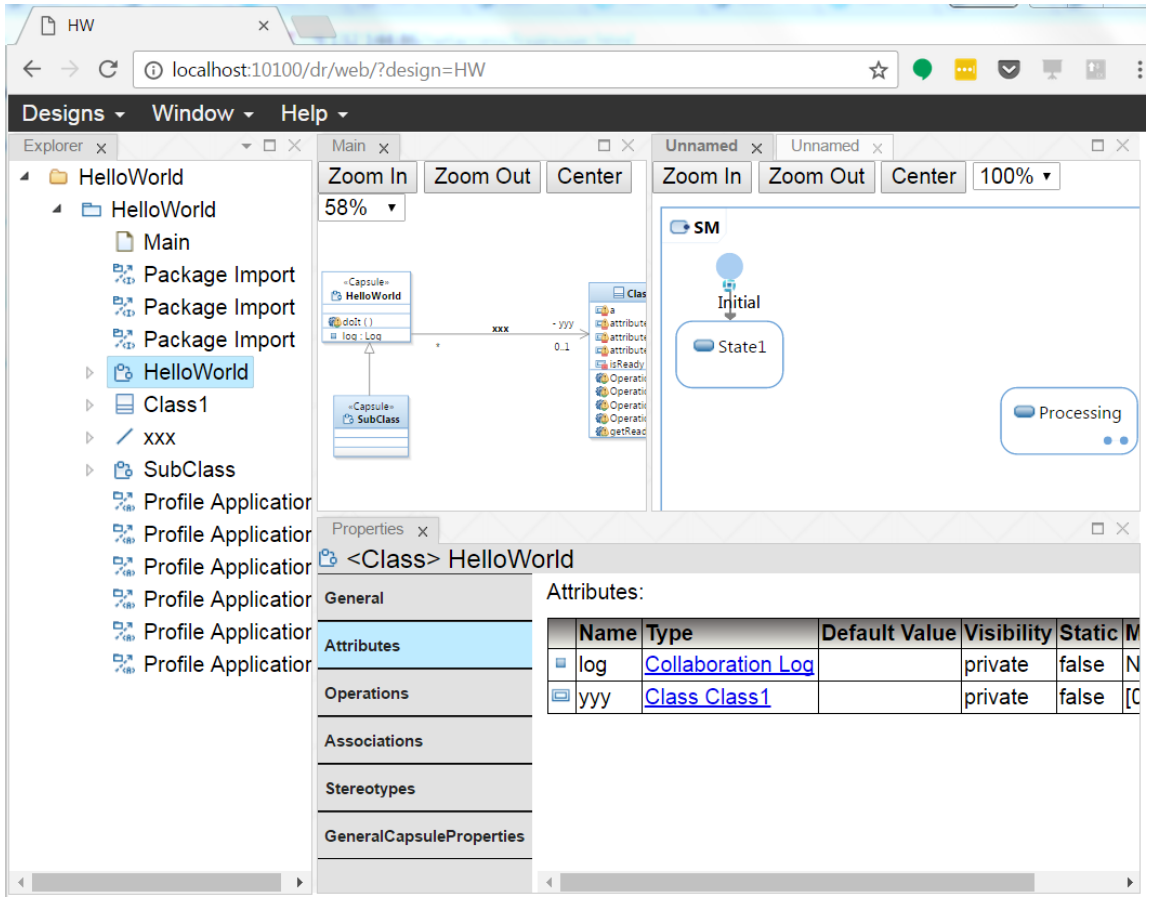
- RSARTE provides a new improved format for transformation configurations
  - Based on JavaScript instead of proprietary format
  - Allows TCs to be dynamic by means of JavaScript code
  - A new editor allows both form-based and code-based editing of TCs
  - Easy to search and compare/merge
- In Rose RT this data is more cumbersome to edit
  - Edited by means of dialogs
  - No possibility to have dynamic build properties





Web Publishing

- RSARTE supports publishing models to a web server
  - A modern interactive web application allows users to browse published models in a web browser
  - The web server supports OSLC, which means links can be created to model elements from artifacts in other OSLC-enabled tools (such as requirements in DOORS NG, work items in RTC, etc.)



- Rose RT generates static HTML files
  - Diagrams are binary images (not possible to search for text within them)
  - No integrated solution for publishing generated files to a web server

Development Team and Process

- RSARTE is actively developed by an experienced team
  - An agile development process is used, where new releases are made available frequently (usually every 3rd week)
  - Actively maintained user community and documentation wiki (https://www.ibm.com/developerworks/connect/rsarte)
  - Development in close cooperation with customers – possible to influence on the future of the tool
- Rose RT is in maintenance mode
  - No updates were made in several years
  - Shrinking user base since customers are now gradually migrating to RSARTE
  - Will eventually run out of support

Migration from Rose RT to RSARTE

- Importing Rose RT models into RSARTE
  - No data is lost when importing, and all code remains unchanged
- Compatible but improved Target RTS and generated code
  - The RSARTE Target RTS is fully compatible with the one in Rose RT
  - Code generated by RSARTE has same structure as in Rose RT, but is more readable and more optimized
- Assistance with a migration effort
  - IBM/HCL has experience from many previous migration projects and can assist both with support and training