
Orocos Real-Time Toolkit 1.2 Release Series Changes, New Features, and Fixes

Open RObot COntrol Software

Table of Contents

1. Caveats	1
1.1. Building the RTT	1
1.2. OS Changes	2
1.3. RTT Changes	3
1.4. Scripting Changes	3
1.5. XML Changes	4
1.6. Corba IDL Changes	4
1.7. Shared Libraries	4
2. Improvements	4
2.1. Shared libraries	4
2.2. RTT classes	5
2.3. Corba Support	5
1. About Orocos	5

This document provides a quick overview of what changed between the Real-Time Toolkit 1.0 and version 1.2. This release includes all bugfixes of the 1.0 branch. If your application does not work, look here for possible causes and solutions. Also feature additions and improvements are documented.

There is a list of tracked issues [https://www.fmtc.be/orocos-bugzilla/buglist.cgi?query_format=advanced&short_desc_type=allwordssubstr&short_desc=&product=RTT&target_milestone=1.2.0&long_desc_type=substring&long_desc=&bug_file_loc_type=allwordssubstr&bug_file_loc=&bug_status=RESOLVED&resolution=FIXED&emailassigned_to1=1&emailtype1=substring&email1=&emailassigned_to2=1&emailreporter2=1&emailcc2=1&emailtype2=substring&email2=&bugidtype=include&bug_id=&votes=&chfieldfrom=&chfieldto=Now&chfieldvalue=&cmdtype=doit&order=Reuse+same+sort+as+last+time&field0-0-0=noop&type0-0-0=noop&value0-0-0=] fixed in this release.

The complete changelog [<http://svn.mech.kuleuven.be/websvn/orocos/trunk/rtt/?op=log&max=150&>] is available as well.

1. Caveats

1.1. Building the RTT

- Orocos RTT is now built with the 'CMake' tool, instead of the automake tool previously. See the installation manual for the new build instructions. The installed libraries and header files names and locations have not changed. However, the pkgconf directory has disappeared and is replaced with the rtt/rtt-config.h file, which contains the same defined macros.

The new build system may on your system detect installed libraries differently than in version 1.0. If you encounter problems, notify the orocos-dev mailinglist and a hotfix will be prepared.

See also [Change Orocos directory structure \[https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=223\]](https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=223) and [\[Project\]Build shared libraries \[https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=28\]](https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=28).

1.2. OS Changes

- Support for real-time PowerPC (40x and 60x were tested) has been added. The functionality has been tested with a Linux 2.4 PPC kernel (patched for Xenomai), with gcc 3.3.6 and glibc 2.2.5 but should work with other kernels and compilers as well.

See also [Rethink the os/oro_*.h assembly functions \[https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=289\]](https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=289).

- The necessary files have been added to run Orocos RTT on eCos. However, you need to create a specially crafted C++ GNU toolchain, see libstdc++ for eCos [<http://www.zylin.com/libstdc++.html>], because the free eCos version does not supply a Standard C++ toolchain. See also [\[Project\] eCos Port \[https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=27\]](https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=27).
- An `rtos_task_yield()` function is now required in the FOSI layer, which maps to the yield function of your OS. If your OS does not support a 'thread yield', you may implement this as an empty function.
- Two functions, `int rtos_task_check_scheduler(int* scheduler)` and `int rtos_task_check_priority(int* scheduler, int* priority)`, have been added to the internal FOSI layer. Each OS must implement these functions and check the arguments for validity.

See also [RTT does not provide means to check scheduler/priority combos \[https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=382\]](https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=382).

- The `OS::SingleThread` and the `NonPeriodicActivity` threading classes had undefined semantics for `start()` and `stop()` with respect to `loop()` in case `start()` was called multiple times in a row. With the old behaviour, `start()` invocations had an accumulative effect, causing `loop()` to be executed as much as `start()` was called, even after `stop()` was called. The new behaviour only allows `start()` to succeed if `loop()` is not executing. `stop()` will only return when `loop()` finishes.

See also [loopActivity](#) [[ht-](#)

[tps://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=253](https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=253)].

1.3. RTT Changes

- See the above remark about the changed behaviour of NonPeriodicActivity.
- The TaskContext startup(), update(), shutdown() user functions are in the process of being renamed. They are still valid, but deprecated and new applications should no longer use these names. The new names are startHook(), updateHook() and stopHook(). The new naming is less confusing in combination with the new configuration state of the TaskContext (see below). Existing RTT 1.0.x applications using the old naming remain working.

See also TaskContext offers no configuration function [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=352].

1.4. Scripting Changes

- The programs.MyProgram and states.MyStateMachine prefixes, required to access loaded programs and statemachines has been dropped. One can now address a TaskContext program or state machine by just writing MyProgram or MyStateMachine in a script. The parser will not allow name clashes between script names and TaskContext objects. The parser has been modified to allow the old syntax and warn the user of the change.

See also [Project]Refactor TaskObject vs TaskContext [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=328].

- Program and state machine scripts are no longer loaded as TaskContext C++ objects but as TaskObject C++ objects. In practice, this has no effect on existing applications, unless the user uses getPeer() in order to access loaded scripts (instead of the recommended states(), programs() or execution()). In that case, the user needs to call getObject() instead of getPeer() or revert to the recommended functions.

See also [Project]Refactor TaskObject vs TaskContext [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=328].

- Toolkit plugins which use the RTT::TemplateTypeInfo class to add user data types to the Real-Time Toolkit must provide operator>> as well for their type when 'use_ostream' is enabled. In other words, if you defined operator<< for your toolkit type, you now also need to define operator>>.

See also Toolkit Plugins should provide read()/operator>> as well [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=301].

1.5. XML Changes

- When a PropertyBag object is stored in an XML struct element, it previously had the type="type_less" attribute. XML files written by RTT 1.2.0 will use the type="PropertyBag" attribute (conforming scripting naming). If an old-style file is found, it is automatically converted.
- When a std::vector<double> property is stored in an XML struct element, it previously had the type type="std::vector<double>," attribute. XML files written by RTT 1.2.0 will use the type="array" attribute (conforming scripting naming). Also, the 'Size' element is no longer required and may be omitted. The elements of the array struct also no longer require a name, and the name is ignored. The ordering in the file counts as index number. If an old-style file is found, it is automatically converted.
- See also [Project]XML format: change of type="xyz" fields [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=326].

1.6. Corba IDL Changes

These changes only apply if you directly used the IDL interface of Orocos.

- The ControlTask class and related classes have been extended to map to the TaskContext and TaskObject interface.

1.7. Shared Libraries

Since the RTT can be build as a shared library, this imposes a limitation on your derived libraries or applications.

- When the RTT is built as a shared library (default), dependent libraries, such as the OCL, KDL or your own library which includes RTT headers, must also be a shared library. If you do not obey this rule, crashes or compilation errors may occur. It is not yet clear if this is a bug in the GNU linker or expected behaviour.

See also Mixing liborocos-rtt.a and .so causes crash upon exit [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=402].

2. Improvements

2.1. Shared libraries

- From release 1.2.0 on, Orocos RTT is built by default as a shared library. This

significantly improves link times and application development times. It also allows dynamic component deployment in the Orocos Component Library.

2.2. RTT classes

- The TaskContext class has been extended to support a configuration state. A component now has a configure() and cleanup() method to configure the component or free all resources. If a component is not configured, it will refuse to start running. The associated user functions (configureHook() and cleanupHook()) allow the user to control the transitions.

See also TaskContext offers no configuration function [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=352].

- The PeriodicActivity and NonPeriodicActivity classes have received additional constructors which allow to set the scheduler type to ORO_SCHED_RT (default) or ORO_SCHED_OTHER. This allows easier creation of not real-time threads in the Orocos RTT.

See also Priority and scheduler of activities [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=314].

- The PeriodicActivity class could no longer be started if its thread was stopped for some reason. If the thread is not running, it tries to start it again.

See also PeriodicActivity does not start PeriodicThread if necessary. [https://www.fmtc.be/orocos-bugzilla/show_bug.cgi?id=367].

2.3. Corba Support

- From release 1.2.0 on, Orocos RTT has better build and run-time support for Corba (using ACE/TAO). Remote modification of component parameters has been improved and is taken advantage of by the Corba based KTaskbrowser 1.2.

1. About Orocos

Please send general, non technical, Orocos questions to orocos at lists.mech.kuleuven.be [[mailto: orocos at lists.mech.kuleuven.be](mailto:orocos@lists.mech.kuleuven.be)].

These pages are maintained by the Orocos team [<http://www.orocos.org/orocos/whatis>].

For questions related to the use of the Orocos Software, please consult these web pages and the Orocos RTT manuals [<http://www.orocos.org/rtt>]. If that fails, the

orocos-dev at lists.mech.kuleuven.be [mailto:orocos-dev at lists.mech.kuleuven.be] mailing list might help. Please send comments on these web pages and the development of Orocos to our developer mailing list at orocos-dev at lists.mech.kuleuven.be [mailto:orocos-dev at lists.mech.kuleuven.be]. All of our lists have public archives [http://lists.mech.kuleuven.be/mailman/listinfo/orocos] (dev public archive [http://lists.mech.kuleuven.be/mailman/listinfo/orocos-dev]).

Copyright (C) FMTC

Verbatim copying and distribution of this entire article is permitted in any medium, provided this notice is preserved.