Release Notes

Thunder R4 (4.3)

Author	Version	Date
Marcel Fransen	0 .1 (Draft)	2023/4/10
Marcel Fransen	0 .2 (Draft)	2023/4/18
Marcel Fransen	1 .0 (Final 4.3)	2023/4/18

Table of Contents

1.Introduction

With the release of Thunder 4.3 enough substantial features and changes were introduced into R4 to warrant providing an overview of these in this release notes document.
For future Thunder R4 minor releases this document will be updated to provide a complete overview for the R4 release as well as a description of the incremental changes between the new and previous minor release.

Please note that R4 does not only encompass changes in the Thunder framework itself but also includes features and changes made to the Thunder plugins in both ThunderNanoServices and RDKServices as well as in the ThunderInterfaces and ThunderClient Libraries repositories. It goes without saying that the features in these repositories mostly build upon features introduced in Thunder R4 and cannot be used with older versions of Thunder. The changes in the plugins are documented in separate documents to be found in the documents folder of the corresponding repository.

For that reason the changes made to Metrological owned plugins in RDKServices have not been introduced in that repository as the whole repository is still based on Thunder R2 (there are no branches for specific Thunder versions). So for now the changes for these plugins can be found in the ThunderNanoServicesRDK shadow repository and when RDKServices moves to R4 will also be integrated there.

2.Thunder changes in R4 (up to release 4.3)

• Message engine

A new message engine was added that replaces the (duplicate) Tracing, Warning Reporting and SysLog reporting engines. This so that all this information is accessible in a consistent way and to increase throughput and performance while on the other side reduce the codesize, so one message channel is used for all three, instead of three separate channels.
This is also very useful for containerization as they are now all in a separate directory.
Please note that the TraceControl and WarningControl plugins are no longer functional for R4 and have been replaced by a single MessageControl plugin.
Note: A beginning with this was already made in R3 but the older reporting engines were still available and selected by default; in R4 however the old engines are removed and the Message Engine replaces them. Also note that of course the interface to report these message categories (so Trace, Syslog and WarningReporting) in code has not changed so no code changes are necessary when switching to the message engine.
As this is a bigger feature, separate documentation for it will be provided to describe this new feature in detail.

• COM/RPC buffer size specification

It is now possible with the “restrict” meta annotation to set the maximum buffer size for a buffer in a COMRPC message. This can also be used for strings (as strings are of course just buffers of characters internally). This way the buffer can be checked for overflow and the total COMRPC message size can now be compressed for smaller sizes as up front it is now known how many bytes are needed to send the size of the buffer (even sizes with a maximum of three bytes are supported).

The “restrict” meta annotation can also be used to limit the range of a parameter. In case a parameter indicates a percentage, the type could be uint8_t and using the restrict annotation, the stub will validate that the value pf the parameter lies between 0-100. If not an error can be returned in the return parameter is a Core::hresult.

- ## JSON RPC over COMRPC

It is now supported to send JSON RPC over COM RPC using the IDispatch interface without the need for serialization. This feature is particularly useful for OOP and Remote plugins.

- ## COMRPC message size improvements

Several improvements have been made to COMRPC reducing the required number of bytes on the wire, improving the efficiency and throughput of COMRPC.

- ## WorkerPool improvements

The Thunder WorkerPool now provides an insight on the Queued jobs, Running jobs and IdleJobs in the WorkerPool. It will report details on these jobs that can be helpful to investigate deadlocks and lack of responsiveness cases. Details will be provided for internal jobs like JSONRPC and COMRPC jobs but also customized jobs can report their details (code changes required to these jobs of course). The details can be seen in the ThunderConsole, can be requested via JSONRPC and also be dumped to file by sending a signal to Thunder or WPEProcess.

- ## User and Group improvements

An OS abstraction was added to enable the specification of Users and Groups for all file handles. This is especially useful for containerized use cases.

- ## JSONRPC Structure support

The JSONRPC generator now supports the usage of POD types (so plain C structs) in the C++ interfaces. These can even be nested (so structs in structs). This is especially useful for cases where bigger numbers of parameters need to be passed. The JSONRPC generator can now parse these structs and their usage in methods and generate a JSONRPC interface for these cases.

- ## Aggregated Plugin support

Thunder now supports Aggregated plugins meaning a plugin that works as a single entry point for one or more plugins hidden behind it (so Parent – child plugins). The child plugin then can be addressed by childcallsign@parentcallsign. As this is a bigger feature, separate documentation for it will be provided to describe this new feature in detail.

- ## ThunderTools

Before R4 the Thunder tooling (tooling developed to be able to build Thunder or some of its components, e.g. ProxyStub generator or JSONRPC code generator) was part of Thunder itself. For consistency and an easier merge/upgrade path these have now been moved into their own ThunderTools repository.

- ## Increased Warning level

The C++ compiler warning level for Thunder has been increased to reduce the chance for possible coding errors. As this of course also leads to false positives a mechanism was introduced to silence these warnings in when desired in a compiler abstract way (so the flag to silence a warning will work for multiple compilers). For code building on top of Thunder the increased warning level is optional and not mandatory. Thunder has been compiled warning free using GCC 11.

- ## FileObserver improvements

The Thunder FileObserver (code mechanism that allows for notification on file content changes in a very resource friendly way, no thread needed, completely embedded into the Thunder infrastructure) is improved to also monitor directories and is now also supported on Windows.

- ## FileOberver on ProxyStub and Configuration Path

Optionally a file observer can be configured on a ProxyStub and Configuration path enabling the dynamic loading of plugins after Thunder was started by providing the configuration and ProxyStub library (in case the Interface was not known at Thunder build time) for this plugin.

[config/observe/proxystubpath]

- ## Support for external plugin root folder

It is now possible to specify in the plugin config that the root file path where the plugin libraries must be loaded is different than the default device root file system, this for improved externally placed/downloaded libraries. Please note this feature due to OS limitations will only work with OOP plugins.

[config/observe/configpath]

- ## JSON RPC Generator enum bitmask support

The JSON RPC generator now also supports the usage of bitmask enums (combination of enum values can be set in the same enum parameter at the same time) in the C++ header file. In the JSON RPC interface this will then be mapped as an array of values. Use the /* bitmask */ annotation for this.

- ## Thunder Singleton improvements

The Thunder singleton implementation was improved. Especially lifetime and parameter passing has seen changes and a pre- and post-init possibilities were added.

- ## Callsign known in thread

In a Thunder thread it is now possible to retrieve the callsign of the plugin the code in this thread is executed for.

- ## Subsystem control improvements

There were multiple subsystem control improvements. It is now possible to set more information using the subsystem interface.
For example it is now possible to specify the latitude and longitude for the device in the plugin implementing the location subsystem

- ## Thunder Hibernation

Thunder now supports Hibernation for plugins, an non cooperative way (from viewpoint of the Plugin) of (partially) removing a plugin from memory while it is not actively used but without deactivating it (so a reduced reactivation time in case the plugin is needed again). At the moment one implementation for a Hibernation library is available.
As this is a bigger feature, separate documentation for it will be provided to describe this new feature in detail.

- ## Security engine improvements

When a JSON RPC request is sent via the controller to the plugin now the correct destination callsign is taken into account for the security check (previously the Controller callsign was used)

- ## Improved plugin library unloading

To resolve some issues with libraries being unloaded while COMRPC calls were still being handled for these libraries in non happy day scenarios, the unloading for Pluging libraries was improved and is now done on a WorkerPool idle job (so also that mechanism is newly introduced).

- ## Performance improvement for internal containers

Some internal data containers were improved leading to improved efficiency and/or reduced memory usage (e.g. using std::vector instead of std::list and std::unorder_map instead of std::map)

- ## JSON Parsing improvements

Several changes to the JSON RPC parser were made, improving its performance and extending its conformance (e.g. improved UTF-8 support).

- ## Improved Plugin failed loading handling

To improve the correct handling of plugins which fail to Initialize when being activated the Deinitialize is now also called when the Initialize fails by the framework so that a proper cleanup of acquired resources can be done symmetrically. So please note that the Deinitialize should always be written in a way it can cleanup everything already acquired during the Initialize. As this might require code changes this behavior can be turned off when it introduces issues. Of course, all Metrological owned plugins have been checked for correctness and adapted when necessary for this.

[config/legacyinitialize]

- ## COMRPC non happy day improvements

A number of improvements have been made to the COMRPC non happy day scenarios (so mostly when the connection would be closed unexpectedly) fixing possible crashes and preventing resource leaks. Also a new callback was added, “Dangling”, that can be implemented to release references still held (e.g. notification/callback sinks) while the connection is terminated unexpectedly.

- ## COMRPC endianness improvements

COMRPC now handles differences in endianness correctly when communicating between two platforms using different endianness.

- ## GITHUB Actions introduction

Thunder (but also other Thunder repositories like ThunderNanoServices) now use Github actions to run multiple builds (debug, release, multiple OS’s) and unit tests when a new Pull Request is ready for merging into the repository. This should prevent merging non compiling code and will fix some issues already before the new code is introduced.

- ## Improved channel logging

When requesting detailed information on open channels (e.g. via the Thunder console or JSON RPC) now more information is available. The list is now complete and with more metadata, for example the destination of the channel and for COMRPC channels which interfaces are open on the channel will be included.

- ## Proxy/Stub generator improvements

Multiple improvements have been made to the COMRPC proxy stub generator. For example const interface pointers are now handled correctly.

- ## Plugin metadata and versioning improvements

Plugins now have a metadata section where the plugin version, subsystem dependencies and which subsystem the plugin implements can be specified.
Also the versioning information available has been improved and now the plugin version as well as the interface version can be retrieved via JSON RPC.

- ## Initialize and Deinitialized notifications

One can subscribe on new notifications when a plugin is being Initialized and being Deinitialiazed. The Initialize notification is called just before the plugin is being activated and for example configuration changes to the plugin being activated are still allowed here. The Deinitialized is called after the plugin Deinitialize has been executed (in contrast to the Deactivated notification which is called before the plugin Deinitialize is called).

3. ThunderTools changes in R4 (up to release 4.3)

• New config generator

As the old plugin configuration generator (the tool that generates an example config for a plugin from some metadata and build flags) was not compatible with newer versions of CMake a new configuration tool was developed which has the advantage, at least to some, that it uses a different, easier to understand meta language. The old config generator is still supported and the new and old can run side by side and in that case the generated configuration files are compared for correctness. The old configuration meta files are to be named .config while the new ones are to be called .conf.in.

The new way of configuration generation is required from CMake version 3.22 and onwards.