5 EXECUTION PLATFORM REQUIREMENTS

This chapter lays out the specific normative requirements on the Execution Platform. The first section presents general requirements relating to the scope, boundaries and structure of the Execution Platform. The second section specifies each of the capabilities required of the Execution Platform, providing detailed requirements for each capability. A final, third, section specifies which Execution Platform capabilities shall be directly accessible to application software.

Requirements are included for all of the capabilities which might be necessary for an Execution Platform in the OSRA. It is expected that this set of requirements will be tailored for application to a specific mission or development to remove unnecessary requirements.

5.1 General Requirements

This section contains requirements relating to the Execution Platform as a whole, and the design approach to be taken to an Execution Platform.

OSRA-EP-GEN-FN-010

General Capabilities

The Execution Platform shall provide the capabilities necessary to permit the execution of application software on a specific hardware platform.

Rationale: This requirement exists to define the general scope and expectations on the Execution Platform.

Comment: The general purpose of the Execution Platform is to permit application software to Execute. Application software is that part of the software which exists to deliver the mission's primary functions and vary depending on mission and system operational concept. The Execution Platform enables this by providing access to avionics, monitoring and control facilities, communications capabilities and the ability to execute software on the underlying hardware. The Execution Platform is not expected to be generic across multiple hardware platforms.

Applicability: ALL

Verification Method: RoD, T

OSRA-EP-GEN-FN-020

Implementation of the Execution Platform

Each OSRA Execution Platform functional capability specified in this document shall be implemented as an architecturally distinct software layer or module.

Rationale: Modularity in the Execution Platform encourages reuse, assists with adaptation to changing requirements and eases collaboration all of which are stated goals of the OSRA. The intention of this requirement is to encourage the designer of an execution platform to follow the outlined architecture. However, if there is an incompatibility with existing implementations, this requirement is already foreseen to be tailored out. This requirement has been placed here, as a single general requirement, rather than split across the requirements set, in order to facilitate non-compliance in a structured way. This permits the use of existing Execution Platform software, which may not be structured according to this requirement, through non-compliance, whilst encouraging new software developments and adaptations to follow a more structured approach better suiting the needs of modern flight software.

Comment: The OSRA Execution Platform functional capabilities are those presented as sub-sections of Section 5.2, specifically:

- *software execution environment;*
- *hardware execution environment;*
- *monitoring and control;*
- *automation;*
- *protocol handling;*
- *on-board communication and device access; and*
- *external system access.*

Applicability: ALL
Verification Method: RoD

5.2 Capabilities Requirements

The functional capabilities of the Execution Platform are broken down into categories, as described in Section 4.2. The same approach is taken to requirements specification within each section, and common terminology is used throughout.

5.2.1 Software Execution Environment

Functions of the Execution Platform which relate to the underlying operating system, supporting libraries and basic software functions are grouped together as the Software Execution Environment. This is distinct from basic functions which relate to the management of the onboard computer platform, which are grouped together as the Hardware Execution Environment in Section 5.2.2.

5.2.1.1 Life Cycle Management

Life Cycle Management functions relate to the initialisation of the system, from the Execution Platform through to applications. This includes the initial application of persistent context information to the system. These requirements also help define the scope of the Execution Platform, which is considered as including the operating system (and hypervisor, if present) but not the bootloader.

OSRA-EP-SWE-FN-010

Entry Point

The onboard software execution entry point shall be located within the Execution Platform.

Rationale: The Execution Platform comprises all low-level elements of the onboard software. The goal of this requirement is to clarify that the entry point shall be in the Execution Platform and not in the application software.

Comment: This is the software entry point from the bootloader, as the bootloader is not considered part of the Execution Platform.

Applicability: ALL
Verification Method: RoD

OSRA-EP-SWE-FN-020

Operating System

Where an Operating System is used, this Operating System shall form part of the Execution Platform.

Rationale: The Execution Platform comprises all low-level elements of the onboard software.

Comment: Whether the operating system is running within a partition (a "guest" operating system), or natively, it is considered part of the Execution Platform.

Applicability: ALL
Verification Method: RoD

OSRA-EP-SWE-FN-030

Hypervisor

Where a hypervisor is used, this hypervisor shall form part of the Execution Platform.

Rationale: The Execution Platform comprises all low-level elements of the onboard software.

Comment: The extent of the Execution Platform runs down to the hardware, and the behaviour and configuration of the hypervisor is considered as within scope. This means that one element of the Execution Platform, the hypervisor, exists outside of any partition; whereas other Execution Platform elements may be present in one or more partitions.

Applicability: TSP
Verification Method: RoD

OSRA-EP-SWE-FN-040

Application Initialisation

The Execution Platform shall give application software an opportunity to initialise after Execution Platform initialisation but before multi-tasking task execution begins.

Rationale: It is important that the system is fully initialised before execution begins. Permitting application initialisation in a single-threaded environment, before the influence of external stimuli, assists in an orderly initialisation.

Comment: In the context of the OSRA, "applications" include both the Interaction and Component Layers. Typically, the Interaction Layer will initialise first, before overseeing the initialisation of components in an order as defined by their dependencies and any constraints on their initialisation order. This requirement can be implemented by providing e.g., a callback mechanism, although other solutions are considered acceptable

Applicability: ALL
Verification Method: T

OSRA-EP-SWE-FN-050

Initialisation Context

The Execution Platform shall ensure that access to context memory (see Section 5.2.1.5) is available during initialisation.

Rationale: To be able to restore correct platform operation in the event of a disruption, fault or failover, applications must have access to information stored in persistent context memory.

Comment: These requirements do not specify the structure of information within the context memory.

Applicability: ALL
Verification Method: T

OSRA-EP-SWE-FN-060

Initialisation Context Identification

As part of the opportunity to initialise, the Execution Platform shall specify to the application software an identifier which may be used to specify the context which should be used for initialisation.

Rationale: It is possible that there could be multiple sets of persisted context information stored in context memory. Additionally, under some circumstances, it could be preferable not to utilise persisted context and to return to some set of hard-coded "factory" defaults. An identifier can be used to distinguish between these cases and can be handled in a consistent way through all OSRA layers.

Comment: These requirements do not specify the mapping between identifiers and contexts. It also does not foresee to specify the value of such identifiers. The data type (numeric, enumeration, file name) or data structure for such identifier is Execution-Platform specific.

Applicability: ALL
Verification Method: T

5.2.1.2 Partition Management

Partition management refers to the control of partition execution state, the management of the partition execution schedule and the provision of inter-partition communication. The functionality specified here draws heavily on the functionality defined as part of ARINC 653 [RD.11][RD.12]. While the requirements definition is clearly inspired by ARINC 653, which is the most known incarnation for time and space partitioning implementations, it is not the intention of this specification to require compliance to the ARINC 653 API.. For example, queuing and sampling ports are prominent communication means in partitioned systems, however other communications means shall be allowed.

OSRA-EP-SWE-FN-110

Partition Status Query

The Execution Platform shall provide a mechanism for a partition to query its own status, and the status of other partitions (providing it has sufficient privileges) including its current operating mode.

Rationale: A partition may need to know its current status in order to make decisions about initialisation and nominal operation. A system partition may need to know the status of other partitions in order to carry out FDIR.

Comment: It is assumed that access to the status of other partitions is controlled by the privileges of a partition which is typically configured at design time. The existence and definition of specific privileges is Execution Platform implementation specific.

Applicability: TSP

Verification Method: T

OSRA-EP-SWE-FN-120

Partition State Control

The Execution Platform shall provide a mechanism for a partition to control its own operating state/mode, and the state/mode of other partitions (providing it has sufficient privileges).

Rationale: A partition may need to be able to control its own mode to accommodate specific operational conditions or for FDIR purposes. A system partition may need to control the mode of other partitions in order to carry out FDIR.

Comment: It is assumed that the ability to control the mode of other partitions is determined by the privileges of a partition which is typically configured at design time. The existence and definition of specific privileges is Execution Platform implementation specific. It is assumed that the initial mode of a partition, on system initialisation, will be configurable at design time.

Applicability: TSP
Verification Method: T

OSRA-EP-SWE-FN-130

Partition Schedule Control

The Execution Platform shall provide a mechanism for a partition to determine the current system execution schedule being used and, providing it has sufficient privileges, to switch the execution schedule to an alternative pre-configured schedule.

Rationale: A partition may need to be able to determine the current schedule in order to modify its behaviour to accommodate specific operational conditions or for FDIR purposes. A system partition may need to know the control the partition execution schedule in order to carry out FDIR.

Comment: It is assumed that all schedules will be defined at design time. It is further assumed that the ability to control the current execution schedule is determined by the privileges of a partition which is typically configured at design time. The existence and definition of specific privileges is Execution Platform implementation specific.

Applicability: TSP
Verification Method: T

OSRA-EP-SWE-FN-140

Queuing Port Communication

The Execution Platform shall provide a mechanism for a partition to communicate with other partitions using queuing ports associated with communication channel such that the partition with the destination queuing port may read messages, the partition with the source queuing port may send messages and either partition may query the port status.

Rationale: Queuing ports are useful for sending messages between partitions in an ordered pattern where messages are queued in a first-in-first-out sequence.

Comment: It is assumed that the existence of the necessary communication channels will be configured at design time, and that should any resource allocation or initialisation be necessary in order to use the queuing port, the Execution Platform will be responsible for this.

Applicability: TSP
Verification Method: T

OSRA-EP-SWE-FN-150

Sampling Port Communication

The Execution Platform shall provide a mechanism for a partition to communicate with other partitions using sampling ports associated with communication channel such that the partition with the destination sampling port may read messages, the partition with the source sampling port may send messages and either partition may query the port status.

Rationale: Sampling ports are useful for sending messages between partitions in an ordered pattern where newer messages overwrite older messages, rather than being queued.

Comment: It is assumed that the existence of the necessary communication channels will be configured at design time, and that should any resource allocation or initialisation be necessary in order to use the sampling port, the Execution Platform will be responsible for this.

Applicability: TSP
Verification Method: T

5.2.1.3 Tasking and Concurrency

The Execution Platform is responsible for providing functionality to be able to execute multiple application software tasks, where each task is a sequential set of software functions. This specification does not require that these tasks be executed concurrently: sequential execution, such as by a single “master loop”, or various cooperative schemes are perfectly acceptable. Where some form of multi-tasking is implemented, the Execution Platform is responsible for providing suitable mutual exclusion and synchronisation mechanisms.

OSRA-EP-SWE-FN-210

Multiple Task Execution

The Execution Platform shall provide a mechanism for executing multiple application software tasks or Execution Platform tasks.

Rationale: Higher layers of the OSRA define their dynamic behaviour in terms of tasks. It is beneficial if this view of the system is consistent across all layers, including the Execution Platform.

Comment: Note that the execution of multiple tasks is not required to be either concurrent or pre-emptive. Sequential cyclical execution, and cooperative multi-tasking systems fulfil this requirement.

Applicability: ALL

Verification Method: RoD

OSRA-EP-SWE-FN-220

Mutual Exclusion

The Execution Platform shall provide mutual exclusion mechanisms so that resources may be protected against multiple access.

Rationale: Mutual exclusion mechanisms are necessary to ensure consistency in accessing shared resources in a system with multiple tasks. These can be semaphores, mutexes, monitors (such as the Ada Protected Object) or any combination of those.

Comment: Note that if concurrent task execution is not supported by the Execution Platform then a mutual exclusion mechanism is not necessary or, whilst conceptually present, is effectively empty.

Applicability: ALL

Verification Method: RoD

OSRA-EP-SWE-FN-230

Synchronisation

The Execution Platform shall provide a synchronisation mechanism to permit one task to notify another task, potentially including additional context information.

Rationale: A synchronisation mechanism is necessary to permit the coordination of activities in a system with multiple tasks. The requirements does not impose a specific implementation mechanism for task synchronisation.

Comment: Note that if concurrent task execution is not supported by the Execution Platform then a synchronisation mechanism is likely to be simple in implementation.

Applicability: ALL

Verification Method: RoD

OSRA-EP-SWE-FN-240

Critical Section

The Execution Platform shall provide a mechanism that guarantees atomic execution of a block of code.

Rationale: Critical sections are necessary to protect a vulnerable code section from inconsistencies, and ensure that once the block of code execution has started it continues uninterrupted, even in the case of processor traps or interrupts.

Comment: Typically implemented by disabling all interrupts.

Applicability: ALL
Verification Method: RoD

5.2.1.4 Error Reporting

Most unrecoverable malfunctions occurring within applications are unlikely to be reported to the Execution Platform. However, provision is made for the Execution Platform to accept error reports from applications permitting the reports to be logged or forwarded to Execution Platform functions or other applications.

The Execution Platform can also be notified of unrecoverable malfunctions by applications, which typically results in the persistent capture of a death report and a system restart or failover. For this reason, this is covered in the scope of the Platform Management in Section 5.3.8.

OSRA-EP-SWE-FN-310

Application Error Reporting

The Execution Platform shall provide a mechanism for applications to report errors, including a severity indication.

Rationale: It is important for applications to be able to indicate errors to the Execution Platform. A severity indication permits the Execution Platform to determine the appropriate response, as per OSRA-EP-SWE-FN-330.

Comment: The mechanism by which an Execution Platform determines the appropriate response to an error is not defined by these requirements and is likely to be Execution Platform-specific.

Applicability: ALL
Verification Method: T

OSRA-EP-SWE-FN-320

Execution Platform Error Reporting

The Execution Platform shall provide a mechanism for reporting errors to applications, including a severity indication.

Rationale: Given the broad scope of the Execution Platform, there are likely to be a number of types of error which are recoverable. In some cases the correct recovery mechanism is likely to be mission-specific, and forms part of high-level behaviour, such as mode management. It is therefore important that applications have the opportunity to handle some Execution Platform errors.

Comment: Which Execution Platform errors are propagated to applications is not defined by these requirements and is likely to be Execution Platform-, or even mission-specific.

Applicability: ALL

Verification Method: T

OSRA-EP-SWE-FN-330

Critical Error Response

The Execution Platform shall have the ability to respond to application errors which indicate a critical severity by restarting/rebooting if necessary (see Section 5.2.2.1).

Rationale: An application error marked as severe could be fatal. The Execution Platform must the ability to handle this.

Comment: This requirement refers specifically to the ability of the Execution Platform to treat errors originating from applications as highly critical or system fatal.

Applicability: ALL

Verification Method: T

OSRA-EP-SWE-FN-340

Error Log

The Execution Platform shall have the ability to log the occurrence of the first N errors, and after this initial storage space if full, then store in a circular buffer the last M errors.

Rationale: Lessons learnt from past operational missions suggest that it is important for investigation reasons to preserve the original sequence of errors, which typically include clues to the root cause of a problem, and avoid them to be overwritten due to lack of log space.

Comment: This can be achieved by implementing a linear buffer of N oldest entries and then a circular buffer with M latest slots.

Applicability: ALL
Verification Method: T

OSRA-EP-SWE-FN-350

Clearing of Error Log

The Execution Platform shall provide to Ground control the capability of clearing error logs.

Rationale: Ground shall have the capability of clearing error logs once their contents is safely downlinked or their assessment is complete.

Comment: This requirement might need to be extended also to application, in case the on-board software implements advanced autonomy. This is however not included in the current version.

Applicability: ALL
Verification Method: T

OSRA-EP-SWE-FN-360

Querying of Error Log

The Execution Platform shall provide the capability for applications to query error logs. This capability shall be restricted to applications having appropriate security privileges, if this notion is implemented on board.

Rationale: Querying error logs can be important for automation support on board. Nevertheless, this shall be restricted to privileged applications.

Comment: This requirement might need to be extended also to application, in case the on-board software implements advanced autonomy. This is however not included in the current version.

Applicability: ALL
Verification Method: T

5.2.1.5 Context Management

It is expected that the Execution Platform should provide a system context facility, which is the ability to store basic information about the current application software state and/or configuration which should be persistent across system restarts. The intention is that, in the event of an unexpected restart, the application software will be able to resume its

operation with limited or no loss of mission functionality. Context information is typically stored in non-volatile safeguard memory.

The view of context memory here is consistent with that specified as part of the SAVOIR Generic OBC Functional Specification [RD.13].

OSRA-EP-SWE-FN-410

Primary Safeguard Memory

The Execution Platform shall provide read and write access to an area of memory whose contents are persistent across hardware and software restarts (e.g. safeguard memory).

Rationale: Access to persistent memory permits orderly recovery on the onboard software from restarts.

Comment: Safeguard memory may be used to provide persistent storage for application context information as well as internal Execution Platform context.

Applicability: ALL
Verification Method: T

OSRA-EP-SWE-FN-420

Secondary Safeguard Memory

The Execution Platform shall provide read and write access to an additional, functionally separate, area of memory whose contents are persistent across hardware and software restarts (e.g. a second area of safeguard memory).

Rationale: The SAVOIR Generic OBC Functional Specification defines two, functionally separate, safeguard memory areas. Functionally separate means that they can be implemented either as physically distinct memory areas or with a same physical implementation but seen as logically / functionally distinct by the software. This requirement ensures that the definition of the Execution Platform is compliant with the OBC functional specification.

Comment: How the two regions of safeguard memory are used to ensure robust context information persistence is Execution Platform-specific and outside the scope of these requirements.

Applicability: ALL
Verification Method: T

OSRA-EP-SWE-FN-430

External System Safeguard Memory

The Execution Platform shall provide read-only access to an additional, functionally separate, area of memory whose contents are persistent across hardware and software restarts and may be written by external systems such as ground (e.g. a third area of safeguard memory).

Rationale: The SAVOIR Generic OBC Functional Specification defines this third, functionally separate, safeguard memory area with the specific proviso of it being read-only to onboard software but writable by external systems. This requirement ensures that the definition of the Execution Platform is compliant with the OBC functional specification.

Comment: The intention of this safeguard memory region is to provide a robust mechanism for external systems, such as ground, to exercise control over the low-level initialisation behaviour of the onboard software. How this region of safeguard memory is used in practice is Execution Platform-specific and outside the scope of these requirements.

Applicability: ALL
Verification Method: T

OSRA-EP-SWE-FN-440

Context Storage

The Execution Platform shall provide the capability of storing in a safeguard memory the context data produced by applications or by the Execution platform.

Rationale: Storing application contexts permits to restart an application with a given context (e.g., application state), for faster convergence after a software or hardware reboot.

Comment: Safeguard memory may be used to provide persistent storage for application context information as well as internal Execution Platform context. The policy for space reserved for context data and the policy to overwrite it is considered as Execution Platform specific.

Applicability: ALL
Verification Method: T

5.2.1.6 Software Libraries

Where a system includes software libraries, including language run-time support, mathematical libraries or other supporting libraries (such as those for compression,

cryptography etc.) those libraries are considered part of the Execution Platform. These requirements help define the scope and content of the Execution Platform.

OSRA-EP-SWE-FN-510

Language Runtime

Where the onboard software implementation programming language requires language run time, the run time shall be considered to be part of the Execution Platform.

Rationale: The Execution Platform comprises all low-level elements of the onboard software. The behaviour and implementation of a language runtime is often critical to the implementation of low-level software and in many cases the Execution Platform could be tightly coupled to a language runtime.

Comment: The coupling between an Execution Platform and language runtime could be dependent on the runtime implementation.

Applicability: ALL

Verification Method: RoD

OSRA-EP-SWE-FN-520

Software Libraries

Where support libraries, not tightly coupled to an application, are available to support high-level functions, these libraries shall be considered part of the Execution Platform.

Rationale: The Execution Platform comprises all low-level elements of the onboard software, this includes any supporting libraries which are not tightly coupled to an application. Example, a mathematical library is likely to be linked directly with an AOCS / GNC application, while higher-level services such as compression, cryptography, etc., are likely to be part of the Execution platform and accessed from it.

Comment: Where a software library is tightly coupled to an application, and especially if it is not shared between different parts of an application, it is permissible to consider that library as being part of the application, rather than the Execution Platform. The intent of this requirement is to make it clear that software libraries do not fall outside the scope of the OSRA. The sentence "be considered part of" also denotes the responsibility for the library, i.e. which lies with the Execution Platform provider and not the application provider. This applies also in the case in which the execution platform library is statically linked to the application.

Applicability: ALL

Verification Method: RoD

5.2.2 Hardware Execution Environment

The Hardware Execution Environment capability relates to the ability to interact with and control core aspects of the avionics. This relates primarily to elements of the onboard computer, or core functionality which may span multiple hardware functions or devices such as access to coordinated or synchronised time and file- or packet-based storage facilities. Not included here are onboard avionics devices beyond the bounds of the computer (and other key systems), which are explicitly addressed in Section 5.2.6.

5.2.2.1 Platform Management

Platform Management relates to the capability to initialise the system and to be able to manage restarts of the underlying computer platform (causing software restart).

OSRA-EP-HWE-FN-010

Board Support Package

Where low-level software is provided to facilitate execution on a specific hardware platform, such as a board support package (BSP), this shall form part of the Execution Platform.

Rationale: The Execution Platform comprises all low-level elements of the onboard software.

Comment: The BSP typically provides support for low-level interfacing to computer hardware and peripherals. As such it forms a key functional part of the Execution Platform.

Applicability: ALL

Verification Method: RoD

OSRA-EP-HWE-FN-020

Memory Management

The Execution Platform shall be responsible for managing memory, including low-level memory allocation.

Rationale: The Execution Platform is responsible for managing all computing resources on behalf of both applications and the Execution Platform itself.

Comment: "Low-level memory allocation" is intended to include the allocation of memory to applications, but not the allocation of memory within an application. The intent of this requirement is to make it clear that any underlying memory management functions do not fall outside the scope of the OSRA.

Applicability: ALL

Verification Method: RoD

OSRA-EP-HWE-FN-030

Memory Protection

Where memory protection and/or spatial partitioning is required, the Execution Platform shall be responsible for providing these functions, using the underlying hardware if necessary.

Rationale: As the Execution Platform is responsible for managing memory, the management of any memory protection mechanisms, such as a hardware memory management or memory protection unit, also falls to the Execution Platform.

Comment: Memory protection is typically a necessary part of TSP systems; however, it is not confined to TSP systems and may be used in a non-partitioned environment. The management of memory protection is expected in to include any part of the computing platform which could access memory, not just the CPU. This would therefore require the management of Direct Memory Access (DMA) controllers by the Execution Platform in accordance with the overall memory protection policy. Note that this requirement does not specify that memory protection must be used by an Execution Platform, even if hardware support is present.

Applicability: ALL
Verification Method: RoD

OSRA-EP-HWE-FN-040

Interrupt Management

The Execution Platform shall be responsible for managing hardware interrupts.

Rationale: The Execution Platform is responsible for managing all parts of the low-level computing platform on behalf of both applications and the Execution Platform itself.

Comment: The intent of this requirement is to make it clear that the management and handling of hardware interrupts does not fall outside the scope of the OSRA.

Applicability: ALL
Verification Method: RoD

OSRA-EP-HWE-FN-050

Compute Resource Management

The Execution Platform shall be responsible for managing access to computing resources, including CPU cores.

Rationale: The Execution Platform is responsible for managing all parts of the low-level computing platform on behalf of both applications and the Execution Platform itself.

Comment: The intent of this requirement is to make it clear that the management and handling of hardware computing resources does not fall outside the scope of the OSRA. This is primarily intended to include CPUs and CPU cores, although any other resource such as co-processors, FPGAs, DSPs, GPUs, would fall under this category.

Applicability: ALL

Verification Method: RoD

OSRA-EP-HWE-FN-060

Execution Time Partitioning

Where CPU execution time partitioning is required, the Execution Platform shall be responsible for providing this function, using the underlying hardware if necessary.

Rationale: As the Execution Platform is responsible for managing computing resources, on a TSP system where these computing resources are allocated to amongst partitions on a temporal basis, the Execution Platform takes responsibility for that.

Comment: The intent of this requirement is to make it clear that the temporal partitioning in a TSP system does not fall outside the scope of the OSRA.

Applicability: TSP

Verification Method: RoD

OSRA-EP-HWE-FN-070

Application Unrecoverable Malfunction Reporting

The Execution Platform shall provide the capability for application software to indicate an unrecoverable malfunction.

Rationale: Application software may identify an unrecoverable malfunction. The ability to report this to the Execution Platform is necessary in order to ensure suitable recovery.

Comment: Although failures may be identified, and handled, within the Execution Platform, depending on the FDIR strategy, it is possible that unrecoverable malfunctions may be identified at application level.

Applicability: ALL

Verification Method: T

OSRA-EP-HWE-FN-080

Unrecoverable Malfunction Handling

The Execution Platform shall provide mechanisms for the definition of the FDIR policy as a result of the detection of an unrecoverable malfunction.

Rationale: An unrecoverable malfunction is one that cannot be recovered in software. A hardware restart is likely required. However, the precise FDIR policy is defined taking into account mission specific requirements.

Comment: The scope of FDIR policy must be chosen appropriately to the operating environment (e.g. mission, hardware, etc.). E.g., in some cases it may be necessary to restart more than one computer.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-090

Unrecoverable Malfunction Death Report

The Execution Platform shall permit application software to include death report information as part of a system fatal error report.

Rationale: A fatal error is one that cannot be recovered in software. A hardware restart is therefore-required.

Comment: Whilst this requirement indicates that a system fatal error shall result in a hardware restart, the scope of the restart must be chosen appropriately to the operating environment (e.g. mission, hardware, etc.). In some cases it may be necessary to restart more than one computer, but this is not required.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-100

Death Report Storage

The Execution Platform shall store death report information in a safe location, persistent across hardware and software restarts (e.g. safeguard memory).

Rationale: Recovery from a fatal error typically involves a restart of software or hardware. Persistent storage of death reports is therefore essential

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-105

Death Report Storage Policy

The Execution Platform shall store the occurrence of the first N errors, and after this initial storage space if full, then store in a circular buffer the last M errors.

Rationale: Lessons learnt from past operational missions suggest that it is important for investigation reasons to preserve the original sequence death reports, which typically include clues to the root cause of a problem, and avoid them to be overwritten due to lack of storage space in case of continuous production of new death reports.

Comment: This can be achieved by implementing a linear buffer of N oldest entries and then a circular buffer with M latest slots.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-110

Death Report Access

The Execution Platform shall provide the ability to query and delete death reports stored from previous fatal errors.

Rationale: To be able to investigate historic fatal errors, access to death report information is essential. As death report storage is finite, it is typically necessary to be able to delete historic death reports.

Comment: This requirement does not specify the manner through which death report information is to be accessed. It also does not require that access to death reports is provided to applications.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-120

Non-Fatal Error Indication

The Execution Platform shall provide the capability for application software to indicate a non-fatal error.

Rationale: It can be useful for the Execution Platform to be aware of non-fatal errors to permit error logging and forwarding of error information to external systems or, in the case of TSP systems, other partitions.

Comment: This requirement places no restrictions on the handling of non-fatal error information by the Execution Platform.

Applicability: ALL

Verification Method: T

OSRA-EP-HWE-FN-130

Non-Fatal Error Information

The Execution Platform shall permit application software to include additional information as part of a non-fatal error report including the degree of severity of the error.

Rationale: Additional error and severity information is useful for diagnosis of non-fatal errors and to permit suitable FDIR.

Comment: This requirement does not specify the size or interpretation of error information, which is left to the Execution Platform implementation. On TSP systems, the Execution Platform may interpret the severity of a non-fatal error to indicate that it is partition-fatal and may act accordingly.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-140

Non-Fatal Partition Error Reporting

The Execution Platform shall provide the capability to indicate to a supervisor partition that a non-fatal error has been generated in another partition.

Rationale: It is typically useful to have the capability for a supervisor partition to be able to supervise one or more other partitions. The reporting of errors indicated by the supervised partitions to the supervisor partition(s) can facilitate this.

Comment: This requirement does not specify the size or interpretation of error information, which is left to the Execution Platform implementation. On TSP systems, the Execution Platform may interpret the severity of a non-fatal error to indicate that it is partition-fatal and may act accordingly.

Applicability: TSP
Verification Method: T

OSRA-EP-HWE-FN-150

Non-Fatal Error Report Access

If the Execution Platform includes facilities for storage of non-fatal error reports the Execution Platform shall provide the ability to query and delete stored non-fatal error reports from previous non-fatal errors.

Rationale: To be able to investigate historic non-fatal errors, access to non-fatal error report information is essential. As storage for non-fatal error reports is likely to be finite, it is necessary to be able to delete historic non-fatal error reports.

Comment: The logging/storage of non-fatal error is not required of the Execution Platform; however, where such functionality is present, this requirement specifies that the storage should be accessible and maintainable.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-160

Software Restart

The Execution Platform shall provide the capability for application software to be able to request a restart and re-initialisation of the software executing on the computer. The Execution platform shall restrict the accessibility of this service to supervisor application software only, if such concept is supported by the Execution platform.

Rationale: Depending on the potential fault, it could be preferable to recover from failure by restarting only the software, rather than the complete hardware computer platform. The right to restart the software shall be restricted to high authority software or, in a TSP environment, to software of supervisor partitions.

Comment: This facility is accessible to application software as it is expected that mission-level FDIR handling will be carried out at application level.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-165

Application or Partition Restart

The Execution Platform shall provide the capability for application software to be able to request a restart and re-initialisation of a single partition, if such concept is supported in the Execution Platform or a single application. The Execution platform shall restrict the accessibility of this service to supervisor application software only, if such concept is supported by the Execution platform.

Rationale: Depending on the potential fault, application / partition criticality and fail-operational constraints, it could be preferable to restarting only a subset of the software, namely a single partition or a single application. The right to restart the software shall be restricted to high authority software or, in a TSP environment, to software of supervisor partitions.

Comment: This facility is accessible to application software as it is expected that mission-level FDIR handling will be carried out at application level.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-170

Hardware Restart

The Execution Platform shall provide the capability for application software to be able to request a restart and re-initialisation of the computer on which the software is running. The Execution platform shall restrict the accessibility of this service to supervisor application software only, if such concept is supported by the Execution platform.

Rationale: Depending on the potential fault, it could be preferable to recover from failure by restarting the complete hardware computer platform on which the Execution Platform is running.
The right to restart the software shall be restricted to high authority software or, in a TSP environment, to software of supervisor partitions.

Comment: This facility is accessible to application software as it is expected that mission-level FDIR handling will be carried out at application level.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-180

Restart Context

As part of a request for restart and re-initialisation, application software shall be permitted to specify an identifier which may be used to specify the context which should be used for the subsequent initialisation.

Rationale: The recovery procedure for a failure, requiring a software or hardware restart, may also require returning the software to a previous "known good" state. This is possible through the use of persistent context. It is therefore important for application software to be able to request that initialisation after a restart applies a specific context.

Comment: This facility is accessible to application software as it is expected that mission-level FDIR handling will be carried out at application level. The management of system context is associated with safeguard memory handling (see requirements in Section 5.2.1.5). The interpretation of identifiers, type (numeric, enumeration, file name) or data structure and their relationship with stored context information, is Execution Platform implementation-specific.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-190

Execution Platform Error Reporting

The Execution Platform shall provide a capability for indicating to the application software that a non-fatal error has occurred in the Execution Platform.

Rationale: Fatal Execution Platform errors result in a restart, which is not immediately visible to applications. On the other hand, as mission-level FDIR is expected to be handled by applications, it is important that non-fatal Execution Platform errors are reported to applications.

Comment: This requirement relates to errors reported asynchronously by the Execution Platform, outside of the scope of Execution Platforms operations which are invoked synchronously by the application software.

Applicability: ALL
Verification Method: T

5.2.2.2 Time Access and Distribution

The Execution Platform is expected to provide access to at least one time source which is assumed to be monotonically increasing. Typically, this is a Spacecraft Elapsed Time (SCET) or Mission Elapsed Time. Additionally, the Execution Platform may provide access to other sources of time information. The synchronisation of time across multiple sources, and the distribution of time to the other elements of the system (beyond the application software) is the responsibility of the Execution Platform.

OSRA-EP-HWE-FN-210

Baseline Time Source Access

The Execution Platform shall provide read access to a monotonically increasing time source by the underlying hardware, if available.

Rationale: Access to a time source is typically very important to applications. Being able to rely on the monotonic nature of the time source simplifies many operations.

Comment: This requirement acts as a baseline, requiring that at least one source of timing is provided. The execution platform leverages on the capabilities of the underlying hardware to provide such time source access. The nature and synchronisation of the time source is not specified by this requirement.

Applicability: ALL

Verification Method: T

OSRA-EP-HWE-FN-220

Baseline Time Source Modification

The Execution Platform shall provide write access to the monotonically increasing time source.

Rationale: It may be necessary for application software to modify the value of the baseline time source.

Comment: This is a separate requirement to that of Baseline Time Access so that it may be tailored out if the capability is not required. It is important to note that, depending on implementation, write access to the time source may violate its monotonic nature.

Applicability: ALL

Verification Method: T

OSRA-EP-HWE-FN-230

Additional Time Source Access

The Execution Platform shall provide read access to additional time sources, where these are present.

Rationale: It is not unusual for onboard software to have access to multiple time sources, this requirement covers read access to those beyond the baseline time source.

Comment: There is no requirement for additional time sources to be monotonic. The nature of additional time sources is Execution Platform implementation-specific. A system with multiple time sources could use e.g., in some phases some internal high-precision oscillators and in other phases an external GNSS as time source, or a specific payload time source.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-240

Additional Time Source Modification

The Execution Platform shall provide write access to additional time sources, where these are present.

Rationale: It is not unusual for onboard software to have access to multiple time sources, this requirement covers write access to those beyond the baseline time source.

Comment: There is no requirement for additional time sources to be monotonic. The nature of additional time sources is Execution Platform implementation-specific.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-250

Time Synchronisation

Where synchronisation between time sources is required, the Execution Platform shall perform the necessary synchronisation.

Rationale: It is not the responsibility of application software to perform time synchronisation.

Comment: Depending on the nature of time synchronisation required, the hardware involved and the dynamic behaviour of the software system, it may well be impossible for time synchronisation to be carried out by applications. Time synchronisation facilities are therefore always located within the Execution Platform for consistency.

Applicability: ALL
Verification Method: T

OSRA-EP-HWE-FN-260

Time Distribution

Where distribution of time value and synchronisation information is required, the Execution Platform shall perform the necessary distribution.

Rationale: It is not the responsibility of application software to perform time distribution.

Comment: This requirement does not define or restrict the mechanism for achieving time distribution. This may be done via dedicated synchronisation lines, such as a Pulse per Second signal, and/or time value distribution over an onboard communication system. It is typically either difficult or impossible for this to be achieved at application layer; therefore, this requirement specifies that time distribution facilities are always located within the Execution Platform for consistency.

Applicability: ALL
Verification Method: T

5.2.2.3 File Access and Management

Where application software or the Execution Platform expects to be able to access and manage files in one or more file stores (either locally or remotely hosted), it is expected that the SAVOIR Data Storage [RD.14] and those facilities will form part of the Execution Platform, rather than application software.

OSRA-EP-HWE-FN-310

File Management Capability

The Execution Platform shall provide a File Management capability compliant to the File Management System specified in the SAVOIR Data Storage System Requirements Document [RD.14].

Rationale: The functionality forming part of the Execution Platform which relates to file system access and management shall be consistent with other SAVOIR standards.

Comment: This requirement defines the functionality which forms part of the file access and management capabilities of the Execution Platform.

Applicability: ALL
Verification Method: RoD, T

OSRA-EP-HWE-FN-320

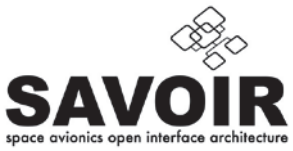
File Management System Location

The implementation of a File Management System shall be part of the Execution Platform.

Rationale: The Execution Platform comprises all storage drivers, file systems and management capabilities.

Comment: The ability to access and manage a file system, where present, is considered to be part of the Execution Platform.

Applicability: ALL
Verification Method: RoD



5.2.2.4 Packet Store Access and Management

The Execution Platform is expected to be able to access and manage packets in one or more packet stores (either locally or remotely hosted). The implementation of such capability in current spacecraft typically leverage the Packet Utilisation Standard. Alternatively, also implementation with files, in a manner compliant with the SAVOIR Data Storage [RD.14] is considered as possible alternative. Irrespectively of implementation, the capability will form part of the Execution Platform, rather than application software. It most cases it is not expected to be appropriate for packet stores to be available to application software, as a packet is an artefact of communication (which is the responsibility of the Execution Platform) and is therefore not visible to application software.

OSRA-EP-HWE-FN-410

Packet Store Management Capability

The Execution Platform shall provide a Packet Store Management capability.

Rationale: The functionality forming part of the Execution Platform which relates to packet store access and management shall be consistent with other standards. Current implementations of this capability leverage on the Packet Utilisation Standard for the Packet Store Capability

Comment: This requirement defines the functionality which forms part of the packet store access and management capabilities of the Execution Platform.

Applicability: ALL
Verification Method: RoD

OSRA-EP-HWE-FN-420

Packet Store Management System Location

The implementation of a Packet Store Management System shall be part of the Execution Platform.

Rationale: The Execution Platform comprises all packet store management capabilities.

Comment: The ability to access and manage a packet store system, where present, is considered to be part of the Execution Platform.

Applicability: ALL
Verification Method: RoD

OSRA-EP-HWE-FN-430

Packet Store Management Capability using Files

The implementation of a Packet Store Management capability, as per requirement OSRA-EP-HWE-FN-410 shall be compliant to the File Management System specified in the SAVOIR Data Storage System Requirements Document [RD.14].

Rationale: The Packet Store Management Capability could be alternatively implemented using files and a files system.

Comment: This requirement defines a packet store management function implemented using files and compliant to [RD.14]. Traditional implementation of the capability leverage on the Packet Utilisation Standard. Those implementation are expected to tailor out this requirement, if not implementing Packet Stores also with file abstractions.

Applicability: ALL

Verification Method: RoD

5.2.3 Monitoring and Control

Monitoring and control is a core activity of the Execution Platform and forms the main interface between the overall platform, including the application software, and external systems. Whilst the communications with External Systems are specified in section 5.2.7, the functions of monitoring and control are specified here. This means that the requirements specified here relate to the monitoring and control functions themselves, as distinct from the reception and transmission of packets relating to these functions. Whilst some implementations may not make this distinction, the conceptual separation is included here both for clarity and to encourage better architected Execution Platform implementations (see Section 5.1).

'Control' is carried out primarily in terms of commandable operations, or *commands*, each of which may be invoked, or executed. Commands may be provided by the Execution Platform itself or by applications. For example, a request to the Execution Platform to enable a monitoring check, or disable the periodic reporting of a set of parameter values are both considered commands.

The invocation of a command may be carried out by an element within the system, such as the Execution Platform or an application, or by an external system via a suitable communications protocol. Commands may, or may not, provide a response; additionally, the execution progress of commands may be available, depending on how they are implemented. The invocation request for a command may include input data. Similarly, the response from a command may be limited to an acknowledgement, including status information, or it may include output data.

'Monitoring' is carried out primarily in terms of *parameters*, items of data of specified types. The different possible types of parameter are not limited here, and may include

vectors/arrays of values and/or structured/record types. Some parameters may be read-only, whilst in other cases it may be possible to set the value of a parameter. As with commands, some parameters may be within the Execution Platform itself, whereas others may reside within applications. The value of a parameter may be acquired, and optionally set (if it is not read-only), by the Execution Platform, applications, and external systems, via a suitable communications protocol. It may also be possible to enable the periodic reporting of sets of parameter values, either on a strict time basis, or when the values have changed. These reports may exist only onboard, to be appropriately logged or archived for later retrieval, or they may be immediately reported to external systems.

Onboard *events* permit the notification of erroneous or notable circumstances within the system. Events may originate from (be emitted by) the Execution Platform or applications. As with parameter value reports, events may be immediately transferred to an external system, via an appropriate communications protocol, and/or they may be logged or archived within the Execution Platform for later retrieval.

Onboard *datasets* are asynchronously emitted collections of data with similar semantics to events. As with events, datasets may originate from (be emitted by) the Execution Platform or applications, may be immediately transferred to an external system, via an appropriate communications protocol, and/or they may be logged or archived within the Execution Platform for later retrieval.

5.2.3.1 Application Commanding

Commands, provided by both applications and the Execution Platform, may be invoked by external systems and by applications.

OSRA-EP-MAC-FN-010

Application Commanding

The Execution Platform shall be capable of invoking commands provided by applications in response to requests from within the Execution Platform, external systems or other applications.

Rationale: Some operations provided by applications can be commandable. In these cases, the Execution Platform shall be able to invoke commands provided by applications. The reason for the invocation may be due to the receipt of a request from an external system, applications, or due to some other functionality within the Execution Platform.

Comment: In the three-layer OSRA, application commands are those provided by both the Interaction and Component Layers. Component Layer commands are expected to be commandable component provided interface operations.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-020

Execution Platform Commanding

The Execution Platform shall be capable of invoking commands provided by the Execution Platform in response to requests from within the Execution Platform, external systems or other applications.

Rationale: The Execution Platform shall be able to invoke its own commands. The reason for the invocation may be due to the receipt of a request from an external system, applications, or due to some other functionality within the Execution Platform.

Comment: This requirement specifically relates to the ability of the Execution Platform to access its own functions as part of M&C handling. When combined with M&C protocol handling this includes the ability for an external system to request the invocation of some Execution Platform functionality by sending a telecommand.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-030

Command Invocation by Applications

The Execution Platform shall be capable of receiving and acting upon command invocation requests from applications.

Rationale: Applications need to be able to request the invocation of commands, independently of the part of the system which provides the command (Execution Platform or application).

Comment: This requirement specifies the generic functionality of command invocation by applications. It does not limit this to commands provided by applications, by the local partition (in TSP systems) or even by the local Execution Platform. The concrete representation of command invocation requests, and their distribution through the system, is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-040

Command Input Arguments

Execution Platform command invocations shall permit the use of input argument values associated with the command.

Rationale: It is common to need to specify input arguments as part of a command invocation request.

Comment: This requirement specifies the generic capability of command invocation requests to have associated input arguments. The concrete representation of command invocation requests, including their input arguments, is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-050

Command Output Arguments

Execution Platform command invocations shall permit the use of output argument values associated with the successful execution of a command.

Rationale: It is common to need to receive output arguments as part of the completion of a command invocation request.

Comment: This requirement specifies the generic capability of command invocation responses to have associated output arguments. The concrete representation of command invocation responses, including their output arguments, is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-060

Command Acknowledgement

Execution Platform command invocations shall permit an acknowledgement of correct or incorrect start of execution of a command, including status information, to be issued by the command provider in response to the invocation request.

Rationale: Command invocation can fail as well as succeed. It is therefore necessary to be able to indicate whether the command invocation led to a successful start of execution.

Comment: This requirement specifies the generic capability of command invocation responses to be able to indicate start of execution status. The concrete representation of command invocation responses, including their status information, is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-065

Command Completion Report

Execution Platform shall support issuing a report on the successful or unsuccessful completion of a command, to be issued by the command provider in response to the invocation request.

Rationale: After a Command has started execution, it can be completed successfully or unsuccessfully. It is therefore necessary to be able to indicate whether the command invocation led to a successful completion of execution.

Comment: This requirement specifies the generic capability of reporting the successful or unsuccessful completion of a command. The concrete representation of command invocation responses, including their status information, is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-070

Command Execution Progress

Execution Platform command invocations shall permit execution progress information to be issued by the command provider in response to the invocation request.

Rationale: In some cases, the execution of a command can take a long time. In these cases it can be useful for the command to be able to provide execution progress information to the invoker.

Comment: This requirement specifies the generic capability of commands to be able to report execution progress information. The concrete representation of command invocation responses, including their status information, is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-075

Reporting of Failure Data after Unsuccessful Command Execution

The Execution Platform shall provide mechanisms to report data, if required, associated to the failure to start, execute or complete a command invocation.

Rationale: Additional data might be necessary to understand the contextual reasons that led to the failure of the command invocation (e.g., syntax mismatch, wrong spacecraft mode, etc...).

Comment: This requirement specifies the generic capability of providing the possibility of reporting additional data after an unsuccessful command execution. The concrete representation of data is not part of this specification and likely require a mapping with an underlying M&C protocol. Implementation might leverage, for example, on output arguments in the original command invocation, or more generically on the capability of reporting unsolicited telemetry contextual to the failure of the command execution.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-080

Command Availability Control

The Execution Platform shall permit the invocation of specific commands to be enabled/disabled such that the attempted invocation of a command that is disabled leads to a negative acknowledgement from the Execution Platform and does not lead to the command being invoked.

Rationale: It can be useful for an Execution Platform to be able to control the availability of specific commands.

Comment: The availability of commands could be determined in response to, for example, the system mode, or in response to a procedure for recovering from a failure.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-090

Command Invocation Transmission

The Execution Platform shall provide mechanisms to communicate the results of command invocation on commandable operations.

Rationale: Acknowledgment, failure and progress reports need to be communicated at least to the command source.

Comment: Important to provide observability on outcome of command invocation.

Applicability: ALL
Verification Method: T

5.2.3.2 Parameter Access and Reporting

Parameters, provided by both applications and the Execution Platform, may be accessed (acquired and set, if permitted) by external systems and applications. Additionally, the values of multiple parameters may be collected and reported for internal archiving or communication to external systems.

OSRA-EP-MAC-FN-110

Requests for Application Parameters

The Execution Platform shall be capable of requesting the value of parameters provided by applications in response to requests from within the Execution Platform, external systems or other applications.

Rationale: The Execution Platform shall be able to request the values of parameters provided by applications. The reason for the request may be due to the receipt of a request from an external system, or due to some other functionality within the Execution Platform.

Comment: In the three-layer OSRA, application parameters are those provided by both the Interaction and Component Layers. Component Layer parameters are expected to be observable component provided interface attributes.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-120

Requests for Execution Platform Parameters

The Execution Platform shall be capable of requesting the value of parameters provided by the Execution Platform in response to requests from within the Execution Platform, external systems or other applications.

Rationale: The Execution Platform shall be able to request the value of its own parameters. The reason for the invocation may be due to the receipt of a request from an external system, applications, or due to some other functionality within the Execution Platform.

Comment: This requirement specifically relates to the ability of the Execution Platform to access its own parameters as part of M&C handling. When combined with M&C protocol handling this includes the ability for an external system to request the value of a parameter related to some Execution Platform functionality by sending a telecommand.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-130

Parameter Request Response

The Execution Platform shall be capable of returning parameter values in response to requests from the Execution Platform to the source of the value request.

Rationale: A successful request for a parameter value should result in the value being returned to the requester.

Comment: This requirement specifically relates to the ability of the Execution Platform to respond to a parameter value request with the value of the parameter. The concrete representation of the parameter request response is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-140

Parameter Request Response Failure

Should the request for a parameter value fail, the Execution Platform shall be capable of returning appropriate status information to the source of the value request.

Rationale: A failed request for a parameter value should result in status information being returned to the requester.

Comment: This requirement specifically relates to the ability of the Execution Platform to respond to a parameter value request with status information. The concrete representation of the response is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-150

Requests for Multiple Parameters

The Execution Platform shall support requests for multiple parameter values.

Rationale: It is common to need the value of several parameters simultaneously. A request permitting multiple parameters to be identified permits the values to be queried at the same time, if this is supported by the implementation.

Comment: This requirement specifically relates to the ability of the Execution Platform to respond to requests for multiple parameter values. The concrete representations of the request and response are not part of this requirement and are covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-160

Requests to Set Application Parameter Values

Where an application parameter is writeable, the Execution Platform shall be capable of setting the value of the parameter in response to requests from within the Execution Platform, external systems or other applications.

Rationale: Some application parameters may be writable (i.e. not read-only) in which case it may be necessary for their value to be set.

Comment: This requirement specifically relates to the ability of the Execution Platform to respond to requests for to set application parameter values. The concrete representation of the request is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-170

Requests to Set Execution Platform Parameter Values

Where an Execution Platform parameter is writeable, the Execution Platform shall be capable of setting the value of the parameter in response to requests from within the Execution Platform, external systems or other applications.

Rationale: Some Execution Platform parameters may be writable (i.e. not read-only) in which case it may be necessary for their value to be set.

Comment: This requirement specifically relates to the ability of the Execution Platform to respond to requests for to set Execution Platform parameter values. The concrete representation of the request is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-180

Parameter Set Request Response Failure

Should the request to set a parameter value fail, the Execution Platform shall be capable of returning appropriate status information to the source of the value request.

Rationale: A failed request to set a parameter value should result in status information being returned to the requester.

Comment: This requirement specifically relates to the ability of the Execution Platform to respond to a parameter value set request with status information. The concrete representation of the response is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-190

Parameter Value Report Definition

The Execution Platform shall support the definition of parameter report specifications, each of which identifies a list of one or more parameters and the required reporting period.

Rationale: Parameter value reports permit the Execution Platform to report the values of multiple parameters asynchronously to requests. This requires that the reports are defined.

Comment: The generation of parameter value reports is covered by requirement OSRA-EP-MAC-FN-210.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-200

Parameter Value Report Definition Modification

The Execution Platform shall support the modification, addition and deletion of parameter report definitions by external systems.

Rationale: It can be necessary to modify the set of parameter report definitions available to the Execution Platform during the operation of the system to accommodate changes in operational requirements or to facilitate diagnosis of issues.

Comment: The generation of parameter value reports is covered by requirement OSRA-EP-MAC-FN-210.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-210

Periodic Parameter Value Report Generation (Housekeeping)

The Execution Platform shall support the periodic generation (i.e., defined by a collection interval) of parameter reports, according to parameter report specifications, which may be retained within the Execution Platform or communicated to an external system.

Rationale: The ability for onboard software to report the values of a set of parameters periodically (housekeeping), with a given collection interval, either for immediate forwarding to external systems or for storage onboard for later forwarding, is frequently a key mission need.

Comment: This requirement specifically relates to the ability of the Execution Platform to report sets of parameter values periodically. The concrete representation of the parameter value reports is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-220

Conditional Parameter Value Report Generation

The Execution Platform shall support the generation of parameter reports, according to parameter report specifications, in response to other conditions such as changes in parameter values.

Rationale: In some cases, such as where parameter values change very infrequently, it can be very inefficient to generate parameter value reports periodically. In these cases it can be more effective to base the generation of parameter value reports on, for example, a change in the value of a parameter.

Comment: This requirement specifically relates to the ability of the Execution Platform to report sets of parameter values in response to aperiodic conditions. The concrete representation of the parameter value reports is not part of this requirement and is covered by requirements on M&C protocol handling (see Section 5.2.5.1).

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-225

Support for super-commutated parameters

The Execution Platform shall support the inclusion of super-commutated parameters in periodic parameter value reports, such that for each super-commutated parameter, N samples are included in each parameter value report.

Rationale: In some cases, it is considered appropriate to perform a super-sampling of given parameters in periodic parameter value reports as specified in OSRA-EP-MAC-FN-210.

Comment: The parameter will be sampled at a sub-period equal to the collection interval divided by N.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-230

Parameter Value Report Generation Control

The Execution Platform shall support the enabling and disabling of parameter report generation.

Rationale: Of the parameter reports defined within the Execution Platform not all may be reported simultaneously as some may only be relevant to specific operational conditions. To avoid the need to remove/add/update report definitions the capability to enable/disable the generation of reports should be available.

Comment: It is expected that enabling/disabling the generation of a parameter value report will not affect the availability or content of a parameter value report definition.

Applicability: ALL

Verification Method: T

5.2.3.3 Avionics Device Data Acquisition and Pooling

Data acquired from avionics devices may be acquired periodically, for example in accordance with a predefined schedule to ensure that onboard communication buses are utilised efficiently. As such, it may be more efficient to collect such data in a "pool" where the latest information can then be retrieved asynchronously to the underlying bus access. Similar situations may arise for other parameter values which are either naturally determined periodically, or require expensive computation and therefore cannot be determined on-demand.

The mention of "avionics device" throughout this section stems from the fact that the Execution Platform scope is primarily platform software. Data acquisition from payloads would also be possible, if they support the same services. The type of data that this

specification aims to is however more oriented to telemetry data rather than bulk data transmission or data streams, or files, which are typical patterns to transmit science data. These services would be then likely limited only to a subset of acquisitions on payloads.

OSRA-EP-MAC-FN-310

Avionics Device Telemetry Acquisition

The Execution Platform shall permit the acquisition of telemetry from avionics devices.

Rationale: Both the Execution Platform and applications may need telemetry data from avionics devices in order to carry out other functions.

Comment: The capability to acquire telemetry data from avionics devices is considered to be part of the Execution Platform.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-320

Avionics Device Telemetry Parameters

Telemetry data acquired from avionics devices by the Execution Platform shall be made available as parameters.

Rationale: Both the Execution Platform and applications may need telemetry data from avionics devices in order to carry out other functions, this is facilitated by making telemetry data available as parameters.

Comment: This requirement refers to the fact that the Execution Platform should, both conceptually and architecturally, treat avionics device telemetry as parameters. This is consistent with all other Monitoring and Control capabilities and ensures that avionics telemetry can be requested (by both applications and external systems, where applicable), reported, monitored, logged etc.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-330

Avionics Device Telemetry Acquisition Trigger

Acquisition of avionic device telemetry data by the Execution Platform shall either be periodic, or on-demand, as required.

Rationale: The trigger for acquiring avionics device telemetry can either be based on time, such that the acquisition is periodic, or carried out on demand in response to, for example, a request for the value of the associated parameter.

Comment: This requirement does not restrict the interpretation of "on-demand".

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-340

Avionics Device Telemetry Pooling

The Execution Platform shall permit telemetry data acquired from avionics devices to be stored for a specified period of time (which may be infinite) during which the Execution Platform may use the stored data rather than requiring an additional acquisition from the device itself.

Rationale: Storing telemetry data acquired from avionics devices and providing that data to the rest of the system as a parameter reduces the number of accesses to the device itself, especially in systems which normally acquire avionics device telemetry on demand.

Comment: In this way the Execution Platform can cache, or pool, telemetry data acquired from devices. This can either be done periodically or on-demand but with a "data lifetime" in the pool. In the former case the pool refreshes periodically, whereas in the latter case the pool refreshes when accessed, only if the data in the pool is "old" (according to the lifetime) thus acting like a cache.

Applicability: ALL
Verification Method: T

5.2.3.4 Parameter Statistics and Monitoring

In many situations it is useful for a spacecraft to be able to monitor the value of telemetry parameters either with a view to responding immediately to value changes, or to remote systems, such as ground. Additionally, as communications with spacecraft may not be continuous, it is often useful to be able to determine various numerical statistics on parameters, such as minimum and maximum values, averages and standard deviations.

OSRA-EP-MAC-FN-410

Parameter Statistics Provision

The Execution Platform shall support the determination and recording of statistics on parameter values including minimum value, maximum value and average value.

Rationale: Parameter minimum/maximum value statistics are frequently useful for identifying off-nominal conditions. Parameter averages are useful to identifying parameter values trends and for permitting range monitoring in situation where a parameter value may have sudden changes (e.g. "spikes") which should be ignored.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-420

Extended Parameter Statistics Provision

The Execution Platform shall support additional parameter value statistics, including minimum delta value, maximum delta value and standard deviation.

Rationale: Minimums/maximums of the rate of change (delta) of a parameter value are sometimes useful for identifying off-nominal conditions. Parameter standard deviations are useful for identifying parameter value trends which can help identify off-nominal conditions or gradual degradation.

Comment: It is expected that these statistics would always be in addition to those specified in requirement OSRA-EP-MAC-FN-410.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-430

Parameter Value Monitoring Check Definition

The Execution Platform shall support the definition of periodic monitoring checks on parameters to determine if their value or their delta falls within specified limits.

Rationale: Parameter monitoring checks permit the Execution Platform to periodically check the exact value, upper or lower bounds or delta associated with a parameter. This requires that the monitoring checks are defined.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-440

Parameter Value Monitoring Check Modification

The Execution Platform shall support the modification, addition and deletion of parameter monitoring check definitions by external systems.

Rationale: It can be necessary to modify the set of parameter monitoring check definitions available to the Execution Platform during the operation of the system to accommodate changes in operational requirements or to facilitate diagnosis of issues.

Comment: The monitoring of parameters in accordance with the defined monitoring checks is covered by requirement OSRA-EP-MAC-FN-430.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-450

Parameter Value Monitoring

The Execution Platform shall support the periodic monitoring of parameter values to determine if their value or their delta falls within specified limits in accordance with the defined monitoring checks; a monitoring check shall change status only if N successive and consistent checks report the same result.

Rationale: Range checks on parameter values or statistics form the basis of monitoring to identify conditions as part of FDIR or triggering other onboard functions. A check shall change status (from successful to failed, or vice versa), only when N successive checks report the same result. This prevents changing status on the occurrence of sporadic spurious readings.

Comment: This requirement concerns only parameter value monitoring. If the Execution Platform is capable of monitoring any statistic derived from the parameter value, this is managed via requirement OSRA-EP-MAC-FN-NEW1.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-460

Parameter Value Monitoring Notification

When a value or a value delta of a parameter being monitored passes from within to outside of the specified limits, the Execution Platform shall emit an onboard event.

Rationale: Onboard events are the most suitable mechanism for identifying asynchronous conditions such as monitoring check failures.

Comment: Onboard events are defined in Section 5.2.3.5. Events are only generated when a subsequent number of checks fail, rather than being generated continuously after, whilst the check is failed. If a parameter value (for example) which has been failing its check returns to the valid range and then, on a subsequent check, leaves the valid range again, a second event would be emitted.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-465

Parameter Statistics Monitoring Check Definition

The Execution Platform shall support the definition of periodic monitoring checks on statistics parameters to determine if their value or their delta falls within specified limits.

Rationale: Parameter statistics monitoring checks permit the Execution Platform to periodically check the statistics value associated with a parameter. This requires that the monitoring checks are defined.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-475

Parameter Statistics Monitoring Check Modification

The Execution Platform shall support the modification, addition and deletion of parameter statistics monitoring check definitions by external systems.

Rationale: It can be necessary to modify the set of parameter statistics monitoring check definitions available to the Execution Platform during the operation of the system to accommodate changes in operational requirements or to facilitate diagnosis of issues.

Comment: The monitoring of parameters in accordance with the defined monitoring checks is covered by requirement OSRA-EP-MAC-FN-NEW1.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-480

Parameter Statistics Monitoring

The Execution Platform shall support the periodic monitoring of parameter statistics to determine if their value or their delta falls within specified limits in accordance with the defined monitoring checks; a monitoring check shall change status only if N successive and consistent checks report the same result.

Rationale: Range checks on parameter statistics are additional monitoring capabilities (in addition to parameter value monitoring) to identify conditions as part of FDIR or triggering other onboard functions. A check shall change status (from successful to failed, or vice versa), only when N successive checks report the same result. This prevents changing status on the occurrence of sporadic spurious readings.

Comment: In addition to being able to monitor the value of a parameter, the Execution Platform may be capable of monitoring any statistic derived from the parameter value. The available statistics are only limited by the tailoring of requirements OSRA-EP-MAC-FN-410 and OSRA-EP-MAC-FN-420.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-485

Parameter Statistics Monitoring Notification

When a parameter statistics being monitored passes from within to outside of the specified limits, the Execution Platform shall emit an onboard event.

Rationale: Onboard events are the most suitable mechanism for identifying asynchronous conditions such as monitoring check failures.

Comment: Onboard events are defined in Section 5.2.3.5. Events are only generated when a subsequent number of checks fail, rather than being generated continuously after, whilst the check is failed. If a parameter value (for example) which has been failing its check returns to the valid range and then, on a subsequent check, leaves the valid range again, a second event would be emitted.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-490

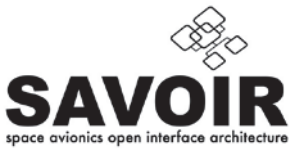
Parameter Monitoring Control

The Execution Platform shall support the enabling and disabling of monitoring checks.

Rationale: Of the parameter monitoring checks defined within the Execution Platform not all may be monitored simultaneously as some may only be relevant to specific operational conditions. To avoid the need to remove/add/update monitoring check definitions the capability to enable/disable specific monitoring checks should be available.

Comment: It is expected that enabling/disabling a monitoring check will not affect the availability or content of a monitoring check, only whether the check is actually carried out. This requirement concerns the capability of control of parameter value monitoring checks and parameters statistics checks, according to whichever is implemented on-board.

Applicability: ALL
Verification Method: T



5.2.3.5 Event Distribution and Reporting

Onboard events can be emitted either by the Execution Platform or applications. Additionally, the Execution Platform and applications may receive events. The Execution Platform is responsible for distributing events from emitters to receivers. Additionally, the Execution may be responsible for the transfer of events to/from external systems.

OSRA-EP-MAC-FN-510

Application Event Emission

The Execution Platform shall support the asynchronous emission of events by applications.

Rationale: The ability to emit onboard events needs to be made available to applications.

Comment: It is expected that there will be a correspondence between events as defined by the Execution Platform, and those defined by applications, including the OSRA Component Layer. The relationship between Execution Platform events and application events is specific to an Execution Platform implementation.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-515

Execution Platform Event Emission

The Execution Platform shall support the asynchronous emission of events by the Execution Platform.

Rationale: The ability to emit onboard events needs to be made available also to the Execution Platform.

Comment: It is expected that there will be a correspondence between events as defined by the Execution Platform, and those defined by applications, including the OSRA Component Layer. The relationship between Execution Platform events and application events is specific to an Execution Platform implementation.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-520

Event Reception by the Execution Platform

The Execution Platform shall support the reception by the Execution Platform of all, or specified, events emitted by applications and the Execution Platform.

Rationale: To support the transfer of events through the system, the Execution Platform must be capable of receiving events emitted by applications and by the Execution Platform itself.

Comment: In addition to providing for the emission of events by applications or itself, the Execution Platform must be capable of receiving them.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-525

Event Reception by Applications

The Execution Platform shall support the reception by applications of all, or specified, events emitted by applications and the Execution Platform.

Rationale: To support the transfer of events to applications, the Execution Platform shall support the reception of events emitted by applications and by the Execution Platform itself.

Comment: The Execution Platform must be capable of supporting the reception of events by applications.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-530

Event Distribution

The Execution Platform shall be capable of distributing events from emitters to receivers.

Rationale: Whilst there is nothing to preclude the distribution of application events within the applications layer, the Execution Platform must be capable of distributing events it receives either from applications or from functions within the Execution Platform.

Comment: The mechanism for event distribution, and its dynamic behaviour, are not specified.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-540

Event Forwarding

The Execution Platform shall be capable of forwarding events to external systems (i.e. as part of the distribution of emitted events to event receivers).

Rationale: As an extension of the event distribution capabilities of the Execution Platform, emitted events can also be forwarded to external systems. How the event is forwarded to external systems and the applicable real-time guarantees for doing so, is considered Execution Platform specific.

Comment: In effect, an external system is acting as an additional event receiver from the point of view of event forwarding.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-550

Event Forwarding Control

The Execution Platform shall be capable of enabling and disabling the distribution of individual and groups of events to external systems.

Rationale: It is likely that only a subset of events will be relevant to an external system. As the relevant events may change under different mission conditions it is important that there is provision for controlling which events are forwarded.

Comment: The manner in which "groups of events" are specified is undefined.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-560

Event Ingestion

The Execution Platform shall be capable of receiving events from external systems and distributing them appropriately within the Execution Platform and to applications.

Rationale: As an extension of the event distribution capabilities of the Execution Platform, events received from external systems can also be distributed to the Execution Platform and applications.

Comment: In effect, an external system is acting as an additional event emitter from the point of view of event distribution.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-570

Event Timestamping

The Execution Platform shall provide a mechanism to timestamp events.

Rationale: Considered useful for detailed investigation of software behaviour.

Comment: Can be useful, especially for investigation of a complex chain of behaviour, or for execution on multi-core targets.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-580

Event Receiver Registration

The Execution Platform shall provide a mechanism to register a new event receiver.

Rationale: Considered useful for registering at run-time an event receiver (i.e., an event handler), possibly only for a specific operation mode.

Comment: Enables flexibility of adding software behaviour in the form of operations to be executed at occurrence of a given event

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-590

Event Receiver Un-registration

The Execution Platform shall provide a mechanism to unregister an event receiver.

Rationale: Considered useful for unregistering at run-time an event receiver (i.e., an event handler), for example, as it is not pertinent for the current operation mode.

Comment: Enables flexibility of removing software behaviour when not appropriate anymore (e.g., after the transition to a new operation mode).

Applicability: ALL
Verification Method: T

5.2.3.6 Dataset Distribution and Reporting

Datasets are short or long data structures exchanged between components (in case of applications) or in general between parts of the Execution platform. This exchange corresponds to a data flow architectural pattern. Contrarily to Events, where it is the occurrence / emission of the event that constitutes the most important element of the communication pattern, for datasets, it is the data that is carried out with the exchange that is primarily important.

The Execution Platform is responsible for distributing datasets from emitters to receivers. Additionally, the Execution may be responsible for the transfer of datasets to/from external systems. Transfer of datasets is performed with the reporting services provided by the Execution Platform[2].

OSRA-EP-MAC-FN-610

Application Dataset Emission

The Execution Platform shall support the asynchronous emission of datasets by applications.

Rationale: The ability to emit onboard datasets needs to be made available to all parts of the software, not just to the Execution Platform.

Comment: It is expected that there will be a correspondence between datasets as defined by the Execution Platform, and those defined by applications, including the OSRA Component Layer. The relationship between Execution Platform datasets and application datasets is specific to an Execution Platform implementation.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-620

Dataset Reception

The Execution Platform shall support the reception of all, or specified, datasets by applications.

Rationale: To support the transfer of datasets through the system, the Execution Platform must be capable of receiving datasets.

---

[2] In case of use of a PUS-based Execution Platform, direct external transfer of datasets is likely to be performed via specific PUS services.

Comment: In addition to providing for the emission of datasets by applications, the Execution Platform must be capable of receiving them.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-630

Dataset Distribution

The Execution Platform shall be capable of distributing datasets from emitters to receivers.

Rationale: Whilst there is nothing to preclude the distribution of application datasets within the applications layer, the Execution Platform must be capable of distributing datasets it receives either from applications or from functions within the Execution Platform.

Comment: The mechanism for dataset distribution, and its dynamic behaviour, are not specified.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-640

Dataset Forwarding

The Execution Platform shall be capable of forwarding datasets to external systems (i.e. as part of the distribution of emitted datasets to dataset receivers) as observable telemetry.

Rationale: As an extension of the dataset distribution capabilities of the Execution Platform, emitted datasets can also be forwarded to external systems. How the dataset is forwarded to external systems and the applicable real-time guarantees for doing so, it is considered Execution Platform specific.

Comment: In effect, an external system is acting as an additional dataset receiver from the point of view of dataset distribution.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-650

Dataset Forwarding Control

The Execution Platform shall be capable of enabling and disabling the distribution of individual and groups of datasets to external systems.

Rationale: It is likely that only a subset of datasets will be relevant to an external system. As the relevant datasets may change under different mission conditions it is important that there is provision for controlling which datasets are forwarded.

Comment: The manner in which "groups of datasets" are specified is undefined.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-660

Dataset Ingestion

The Execution Platform shall be capable of receiving datasets from external systems and distributing them appropriately within the Execution Platform and to applications.

Rationale: As an extension of the dataset distribution capabilities of the Execution Platform, datasets received from external systems can also be distributed to the Execution Platform and applications.

Comment: In effect, an external system is acting as an additional dataset emitter from the point of view of dataset distribution.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-670

Dataset Timestamping

The Execution Platform shall provide a mechanism to timestamp datasets.

Rationale: Considered useful for detailed investigation of software behaviour.

Comment: Can be useful, especially for investigation of a complex chain of behaviour, or for execution on multi-core targets.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-680

Dataset Receiver Registration

The Execution Platform shall provide a mechanism to register a new dataset receiver.

Rationale: Considered useful for registering at run-time a dataset receiver, possibly only for a specific operation mode.

Comment: Enables flexibility of adding software behaviour / processing in the form of operations to be executed at reception of a given dataset

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-690

Dataet Receiver Un-registration

The Execution Platform shall provide a mechanism to unregister a dataset receiver.

Rationale: Considered useful for unregistering at run-time a dataset receiver, for example, as it is not pertinent for the current operation mode.

Comment: Enables flexibility of removing software behaviour / processing when not appropriate anymore (e.g., after the transition to a new operation mode).

Applicability: ALL

Verification Method: T

5.2.3.7 Onboard Storage and Retrieval

External systems, such as ground-based systems, are frequently out of contact with onboard systems. Additionally, the bandwidth of communications links to/from external systems is frequently limited. For these reasons it is often necessary to permit the storage and archiving of information within the Execution Platform for later retrieval by external systems. It is expected that onboard storage and retrieval concerns primarily observable data products, i.e., that are declared in the spacecraft TM/TC ICD.

The Execution Platform provides means to activate and deactivate storage. It is left to implementation and mission specific requirements to specify how and when onboard storage contents shall be deleted.

OSRA-EP-MAC-FN-710

Command Invocation Storage

The Execution Platform shall be capable of storing command invocation, execution and progress information for later retrieval by external systems.

Rationale: Contact with onboard software is frequently limited, therefore the storage and archiving of command invocation is typically essential for verifying operation and diagnosing issues.

Comment: The concrete representation of the information stored in the storage (e.g., packet store) is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-720

Parameter Value Report Storage

The Execution Platform shall be capable of storing parameter value reports for later retrieval by external systems.

Rationale: Contact with onboard software is frequently limited, therefore the storage of parameter value reports is typically essential for verifying operation and diagnosing issues.

Comment: The concrete representation of the information stored in the storage (e.g., packet store) is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-730

Event Storage

The Execution Platform shall be capable of storing onboard events for later retrieval by external systems.

Rationale: Contact with onboard software is frequently limited, therefore the storage of onboard events is typically essential for verifying operation and diagnosing issues.

Comment: The concrete representation of the information stored in the storage (e.g., packet store) is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-740

Dataset Storage

The Execution Platform shall be capable of logging onboard datasets for later retrieval by external systems.

Rationale: Contact with onboard software is frequently limited, therefore the storage of onboard datasets could be useful for verifying operation or diagnosing issues.

Comment: The concrete representation of the information stored in the storage (e.g., packet store) is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-750

Storage Time Recording

The Execution Platform shall be capable of recording the time at which any item (command invocation, status or progress, parameter value report or event) is stored.

Rationale: Recording the time of a storage entry is essential for being able to reconstruct historic information.

Comment: The time source used for logging is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-760

Time-Based Storage Retrieval

The Execution Platform shall support the retrieval of stored information on the basis of time.

Rationale: Retrieving historic storage information on the basis of time assists with reconstructing and maintaining an understanding of the historic behaviour of the system.

Comment: The criteria by which the time may be specified is left undefined but typically this would include all records since, all records until and all records with a specified range. The use of time in specifying storage entries to retrieve is not expected to be mutually exclusive with the use of storage type (as specified by requirement OSRA-EP-MAC-FN-770).

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-770

Type-Based Storage Retrieval

The Execution Platform shall support the retrieval of logged information on the basis of item type (command invocation, status or progress, parameter report or event).

Rationale: Retrieving historic log information on the basis of type assists with reconstructing and maintaining an understanding of the historic behaviour of the system.

Comment: The use of type-based storage to retrieve entries is not expected to be mutually exclusive with the use of time (as specified by requirement OSRA-EP-MAC-FN-760).

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-780

Deactivation of onboard storage

The Execution Platform shall provide the capability to deactivating onboard storage. This capability shall be restricted to applications having appropriate security privileges, if this notion is implemented on board.

Rationale: In order to cope with certain mission or failure scenarios, it might be necessary to deactivate on-board storage.

Comment: Execution platforms without security segregation are expected to tailor out this requirement.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-790

Activation of onboard storage

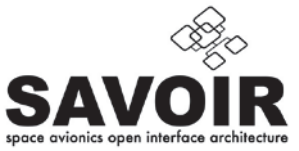
The Execution Platform shall provide the capability to activating onboard storage. This capability shall be restricted to applications having appropriate security privileges, if this notion is implemented on board.

Rationale: Requirement permitting to mirror capability OSRA-EP-MAC-FN-NEW12.

Comment: Execution platforms without security segregation are expected to tailor out this requirement.

Applicability: ALL

Verification Method: T



5.2.3.8 Onboard Logging and Retrieval

External systems, such as ground-based systems, are frequently out of contact with onboard systems. Additionally, the bandwidth of communications links to/from external systems is frequently limited. Logging and archiving of information within the Execution Platform for later retrieval by external systems is considered as a mechanisms capable of providing valuable information for debugging on ground. Onboard logging is in general defined as a capability complementary to Onboard Storage and Retrieval. Whilst Onboard Storage and Retrieval concerns primarily observable data products (i.e., declared in the spacecraft TM/TC ICD), onboard logging concerns any on-board parameter, data products, but also code variables, specific conditions, which could help to perform more in-depth debugging of onboard execution.

The Execution Platform provides means to activate and deactivate logging. It is left to implementation and mission specific requirements to specify how and when logged contents shall be deleted.

OSRA-EP-MAC-FN-810

Onboard logging of observable data products

The Execution Platform shall be capable of storing observable data products for later retrieval by external systems.

Rationale: Logging can be used to log the same data products as onboard storage, although the logging implementation could have more flexible or specialised way of singling out the desired data products.

Comment: The concrete representation of the information stored in the log is not defined.

Applicability: ALL

Verification Method: T

OSRA-EP-MAC-FN-820

Onboard logging

The Execution Platform shall be capable of storing code functions invocation, parameters, variable values, for later retrieval by external systems.

Rationale: Logging can be used to perform detailed and fine-grained debugging of on-board execution, with also data not necessarily declared in the spacecraft TM/TC ICD .

Comment: The concrete representation of the information stored in the log is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-830

Deactivation of onboard logging

The Execution Platform shall provide the capability to deactivating logging. This capability shall be restricted to applications having appropriate security privileges, if this notion is implemented on board.

Rationale: In order to cope with certain mission or failure scenarios, it might be necessary to deactivate on-board logging.

Comment: Execution platforms without security segregation are expected to tailor out this requirement.

Applicability: ALL
Verification Method: T

OSRA-EP-MAC-FN-840

Activation of onboard logging

The Execution Platform shall provide the capability to activating onboard logging. This capability shall be restricted to applications having appropriate security privileges, if this notion is implemented on board.

Rationale: Requirement permitting to mirror capability OSRA-EP-MAC-FN-NEW14.

Comment: Execution platforms without security segregation are expected to tailor out this requirement.

Applicability: ALL
Verification Method: T

5.2.4 Automation

Execution Platform automation features permit the specification of commands, or sets of commands, which should be carried out by the Execution Platform not in direct response to a command from and external system (such as ground) but in response to some stimulus within the Execution Platform. Examples of such stimuli for triggering onboard commands are:

- the current absolute time;the current position of a physical system (such as the spacecraft hosting the Execution Platform), or time relative to a specific position;
- an event being handled by the Execution Platform.

Additionally also sequences of commands are supported. These are triggered on explicit command (on-board or from an external system), and then proceed autonomously. The trigger of each command in the sequence is:

- the “start” command for the sequence for the first operation
- the relative time from the release of the previously executed command in a sequence;

These requirements use the term *command* in the context of automation to refer to a software activity which may be initiated by automation. What constitutes an command, in this context, is left deliberately unspecified by these requirements and should be specified by mission or project requirements.

5.2.4.1 Time-Based Automation

Time-based automation is the initiation of commands on the basis of either absolute time, or the relative time since either a previous command in a sequence, or a specific command to start a relative time sequence.

OSRA-EP-AUT-FN-010

Absolute Time Schedule Specification

The Execution Platform shall support the specification of sequences of commands, where each command is associated with an absolute time indicating when that command should be initiated. This sequence of absolutely-timed commands is an onboard schedule.

Rationale: Defining schedules of commands in terms of the absolute time at which they will be executed is typically an important part of automated commands.

Comment: The time source to be used as a reference for the absolute times specified is not defined by this requirement.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-020

Multiple Absolute Time Schedules

The Execution Platform shall permit the concurrent specification of multiple absolute time command sequences (schedules).

Rationale: It is frequently useful to be able to define multiple schedules for different purposes.

Comment: This requirement does not specify that all concurrently-specified schedules should necessarily be considered for execution concurrently. There may only be one active schedule at a time.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-030

Absolute Time Schedule Time Source

The Execution Platform shall clearly identify the source of time to be used for the timing of absolute time command sequences (schedules).

Rationale: An Execution Platform may have access to multiple time sources. It is important that the time schedule to be used for the execution of an absolute time schedule is well defined.

Comment: This requirement does not limit the Execution Platform to selecting a single time source for schedule execution, just that the time source should be identified and well-known.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-040

Absolute Time Schedule Removal

The Execution Platform shall support the removal of absolute time command sequences (schedules).

Rationale: Onboard storage space for schedules is finite and therefore it is likely that schedules will need to be removed after their execution. Additionally, it may be necessary to remove a schedule before it has been executed should operational requirements change.

Comment: These requirements do not specify that schedules should continue to exist after their execution; however, this requirement ensures that schedules can be removed either before or after their execution.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-045

Absolute Time Schedule Modification

The Execution Platform shall support the modification of absolute time command sequences (schedules).

Rationale: It may be necessary to modify a schedule before it has been executed should operational requirements change. These modifications include e.g., addition of new commands, removal of commands from schedule, modification of time.

Comment: The modification capabilities are defined by the Execution Platform implementation.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-050

Absolute Time Schedule Execution

The Execution Platform shall support the execution of absolute time command sequences (schedules) which results in the initiation of each command in the sequence at the specified absolute time, if the command is enabled.

Rationale: Commands in a schedule should be initiated at the time specified by the schedule.

Comment: This requirement states that commands should always be initiated at the time specified by the schedule. It makes no specific exception in relation to the execution of previous commands in the schedule, which may be relevant to an implementation. It may therefore be necessary to tailor this requirement depending on the mission requirements.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-060

Absolute Time Schedule Execution Progress

The Execution Platform shall permit the execution progress of an absolute time command sequence to be observed and monitored by other parts of the Execution Platform and by external systems.

Rationale: Automated commands should always be observable to ensure that the state of the system can be determined.

Comment: The mechanism for the reporting of schedule execution progress is not defined.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-070

Absolute Time Schedule Execution Control

The Execution Platform shall permit absolute time command sequences to be enabled/disabled such that only an enabled command sequence is considered for execution, whereas a disabled command sequence is not.

Rationale: To accommodate changes in operational conditions or requirements, such as a change in system mode, it may be necessary to be able to enable/disable a schedule.

Comment: This requirement defines a mechanism for enabling and disabling a complete schedule. The enabling/disabling of individual operations in a schedule is not defined.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-075

Absolute Time Schedule Deletion Criteria

Whenever the absolute time for execution of a command has passed, the command shall be deleted from the absolute time schedule.

Rationale: Commands specified for absolute time execution do not have any more meaningful presence in the absolute time schedule. They have been either executed (if enabled) or skipped (if disabled).

Comment: This requirement clarifies the behaviour related to possible late enabling activation of commands.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-080

Relative Time Sequence Specification

The Execution Platform shall support the specification of sequences of commands, where each operation is associated with a relative time indicating when that command should be initiated relative to the release of the last command in the sequence.

Rationale: Defining sequences of commands in terms of the relative time at which they will be executed is typically an important part of automated commands.

Comment: The time source to be used as a reference for the absolute times specified is not defined by this requirement.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-090

Multiple Relative Time Sequences

The Execution Platform may permit the concurrent specification of multiple relative time command sequences.

Rationale: It is frequently useful to be able to define multiple operational sequences for different purposes.

Comment: This requirement does not specify that all concurrently-specified sequences should necessarily be considered for execution concurrently. There may only be one active sequence at a time.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-100

Relative Time Sequence Time Source

The Execution Platform shall clearly identify the source of time to be used for the timing of relative time command sequences.

Rationale: An Execution Platform may have access to multiple time sources. In terms of synchronisation with other onboard operations, it is important that the time source to be used for the execution of relative time operational sequence is well defined.

Comment: This requirement does not limit the Execution Platform to selecting a single time source for sequence execution, just that the time source should be identified and well-known.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-110

Relative Time Sequence Removal

The Execution Platform shall support the removal of relative time command sequences.

Rationale: Onboard storage space for operational sequences is finite and therefore it is likely that sequences will need to be removed after their execution. Additionally, it may be necessary to remove a sequences before it has been executed should operational requirements change.

Comment: These requirements do not specify that sequences should continue to exist after their execution; however, this requirement ensures that sequences can be removed either before or after their execution.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-115

Relative Time Schedule Modification

The Execution Platform shall support the modification of relative time command sequences (schedules).

Rationale: It may be necessary to modify a schedule before it has been executed should operational requirements change. These modifications include e.g., addition of new commands, removal of commands from schedule, modification of time offsets.

Comment: The modification capabilities are defined by the Execution Platform implementation.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-120

Relative Time Sequence Execution

The Execution Platform shall support the execution of relative time command sequences which results in the initiation of each command in the sequence at the specified time relative to the release of the previous command, with the first command in the sequence being initiated in response to an explicit command.

Rationale: Commands in a sequence should be initiated at the time implied by the relative times in the operational sequence.

Comment: This requirement states that operations should always be initiated at the time specified by the sequence. It makes no specific exception in relation to the execution of previous commands in the sequence, which may be relevant to an implementation. It may therefore be necessary to tailor this requirement depending on the mission requirements.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-130

Relative Time Sequence Execution Progress

The Execution Platform shall permit the execution progress of a relative time command sequence to be observed and monitored by other parts of the Execution Platform and by external systems.

Rationale: Automated commands should always be observable to ensure that the state of the system can be determined.

Comment: The mechanism for the reporting of operational sequence execution progress is not defined.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-140

Relative Time Sequence Execution Control

The Execution Platform shall permit relative time command sequences to be enabled/disabled such that only an enabled command sequence is considered for execution, whereas a disabled operation sequence is not.

Rationale: To accommodate changes in operational conditions or requirements, such as a change in system mode, it may be necessary to be able to enable/disable a sequence.

Comment: This requirement defines a mechanism for enabling and disabling a complete operational sequence. The enabling/disabling of individual operations in a sequence is not defined.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-150

Relative Time Sequence Abort

Should an executing relative time command sequence be disabled then execution of the sequence shall be aborted meaning that execution does not resume if the sequence is re-enabled unless a further explicit command is received to re-start the sequence execution.

Rationale: The commands in a sequence typically identify a coherent set. It is therefore important that re-enabling a sequence does not result in the resumption of execution as the context for the execution of the sequence may have changed and may no longer be relevant.

Comment: This requirement does not preclude the addition of a pause/resume mechanism for relative time operational sequences.

Applicability: ALL
Verification Method: T

5.2.4.2 Position-based Automation

Position-based automation is the initiation of commands on the basis of the current, past or future physical position of an object of interest. In most cases this will be the spacecraft which is hosting the Execution Platform, but this does not have to be the case.

OSRA-EP-AUT-FN-210

Position Sequence Specification

The Execution Platform shall support the specification of sequences of commands, where each command is associated with a physical position indicating where that command should be initiated.

Rationale: Defining schedules of commands in terms of the absolute physical position of the spacecraft at which they will be executed is frequently a requirement for automated operations.

Comment: Neither the source of position information, nor its expression format or reference frame is defined by this requirement. It is expected that the frame of reference will relate to the physical body containing the computer which is hosting the Execution Platform; however, this does not have to be the case.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-220

Position Sequence Specification with Timing

The Execution Platform shall permit each step in the position command sequence to additionally specify a relative, positive time, indicating that the command should be initiated at the specified time relative to the time at which the physical position is reached.

Rationale: It is frequently desirable to relate a specific command, especially a payload operation, to a physical location. Activities which may need to occur after this operation are frequently expressed in terms of time.

Comment: An example of this would be a requirement for performing a sequence of image capture operations after reaching a location at which an image must be taken. The time source for the timing is not specified by this requirement.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-225

Position Sequence Specification with Timing

The Execution Platform shall permit each step in the position command sequence to additionally specify a relative, negative time, indicating that the command should be initiated at the specified time prior to the time at which the physical position is reached.

Rationale: It is frequently desirable to relate a specific command, especially a payload operation, to a physical location. Activities which may need to occur before this operation are frequently expressed in terms of time.

Comment: An example of this would be a requirement for powering on an imaging payload two minutes prior to the location at which the image must be taken. The time source for the timing is not specified by this requirement. Complexity of implementation of this requirement on board justifies it as a separate requirement w.r.t. OSRA-EP-AUT-FN-220.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-230

Position Sequence Command Repetition

The Execution Platform shall permit each command in the position operation schedule to be marked as either one-shot or persistent.

Rationale: An orbital spacecraft will revisit the same physical locations. It is therefore useful to be able to identify whether a given command should be carried out only the first time the location is visited, or whether it should be carried out every time.

Comment: The specification of repeat/on-shot applies to individual sequence command, rather than to the complete schedule.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-240

Multiple Position Command Sequences

The Execution Platform shall permit the concurrent specification of multiple position command sequences.

Rationale: It is frequently useful to be able to define multiple sequences for different purposes.

Comment: This requirement does not specify that all concurrently-specified sequences should necessarily be considered for execution concurrently. There may only be one active sequence at a time.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-250

Position Sequence Position and Time Source

The Execution Platform shall clearly identify the source of position and frame of reference to be used for the determination of location for the initiation of commands in a position command sequence, and the source of time for the specification of time offsets.

Rationale: An Execution Platform may have access to multiple position information sources, potentially relating to multiple frames of reference. It is important that the position information source, and its accompanying frame of reference, to be used for the execution of a position command sequence is well defined.

Comment: It is expected that the time source will be correlated with the position source; for example, if a GNSS system is used as the source of position information, time information is likely to be drawn from the same source. This requirement does not limit the Execution Platform to selecting a single position/time source for sequence execution, just that the position information source should be identified and well-known.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-260

Position Sequence Removal

The Execution Platform shall support the removal of position command sequences.

Rationale: Onboard storage space for sequences is finite and therefore it is likely that sequences will need to be removed after their execution. Additionally, it may be necessary to remove a sequence before it has been executed should operational requirements change.

Comment: These requirements do not specify that sequences should continue to exist after their execution; however, this requirement ensures that sequences can be removed either before or after their execution.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-270

Position Sequence Execution

The Execution Platform shall support the execution of position command sequences which results in the initiation of each command in the sequence when the physical position is reached, subject to the relative positive offset (if specified).

Rationale: Commands in a position operation sequence should be initiated at the position/time specified by the schedule.

Comment: This requirement states that operations should always be initiated at the position (plus positive time offset, if relevant) specified by the sequence. It makes no specific exception in relation to the execution of previous operations in the sequence, which may be relevant to an implementation. It may therefore be necessary to tailor this requirement depending on the mission requirements.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-275

Position Sequence Execution before reaching position

The Execution Platform shall support the execution of position command sequences which results in the initiation of operation in the sequence with negative offset w.r.t. the physical position being reached.

Rationale: Commands in a position operation sequence should be initiated at the position/time specified by the schedule. For negative offsets, this implies an implementation capable of predicting that a given position will be reached in the future.

Comment: This requirement states that operations with a negative offset should be initiated in advance w.r.t. the target position. The capability of predicting the spacecraft will reach a given position in the future may be demanding on implementation. It may therefore be necessary to tailor out this requirement if not foreseen by mission requirements.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-280

Position Sequence Execution of One-Shot Commands

During execution of a position command sequence the Execution Platform shall disable any step marked as one-shot once the associated command has been initiated in order to prevent further initiation of that operation even if the time/position conditions necessary to cause command initiation are subsequently met for a second time.

Rationale: Sequence commands marked as one-sot should only execute once for a given execution of a sequence.

Applicability: ALL

Comment: It is assumed that a sequence must be explicitly reset to return the state of one-shot commmand execution to "un-executed".

Verification Method: T

OSRA-EP-AUT-FN-290

Position Sequence Execution Progress

The Execution Platform shall permit the execution progress of a position command sequence to be observed and monitored by other parts of the Execution Platform and by external systems.

Rationale: Automated operations should always be observable to ensure that the state of the system can be determined.

Comment: The mechanism for the reporting of sequence execution progress is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-300

Position Sequence Execution Control

The Execution Platform shall permit position command sequences to be enabled/disabled such that only an enabled command sequence is considered for execution, whereas a disabled command sequence is not.

Rationale: To accommodate changes in operational conditions or requirements, such as a change in system mode, it may be necessary to be able to enable/disable a position command sequence.

Comment: This requirement defines a mechanism for enabling and disabling a complete sequence. The enabling/disabling of individual commands in a sequence is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-310

Position Sequence Execution Reset

The Execution Platform shall permit position command sequences to be reset, such that all one-shot sequence steps are re-enabled, and considered for initiation again.

Rationale: One-shot operations in a sequence will only execute once. To permit a sequence to be re-used it is necessary to reset one-shot commands to permit them to be executed again.

Comment: These requirements treat the enabling a position command sequence as distinct from reset. An implementation may choose to connect these functions, if that fits better with the implementation or mission application.

Applicability: ALL
Verification Method: T

5.2.4.3 Event-Based Automation

Event-based automation is the release of commands in response to the receipt of onboard events. Associations between events and commands may be treated individually, or in sets. The automation service has visibility only on commands (as they are included in the spacecraft TM/TC ICD) rather than on all operations. For this reasons the service associates events to commands, and not events to operations.

OSRA-EP-AUT-FN-410

Event-Command Specification

The Execution Platform shall support the specification of event-command pairs which associate the release of the specified command with the receipt of the specified event.

Rationale: Being able to respond to the emission of onboard events, for example due to the failure of a monitoring check, typically forms an important part of onboard automation, such as response to faults. It shall be noted that the operation initiated by the event-command mechanism can be the trigger command for a relative-time sequence, as defined in section 5.2.4.1. This offers the possibility to trigger a sequence of operations at reception of a single event.

Comment: This requirement treats event-command pairs individually rather than in sets.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-420

Event-Command Set Specification

The Execution Platform shall support the association of groups of event-command pairs to form event-operation pair sets such that an event-command pair is associated with one event-command set.

Rationale: It can be useful to work with groups of event-command pairs, rather than the individual pairs.

Comment: This permits sets of event-command pairs to be associated with, for example, particular system states or modes of operation. The intention is that each event-command pair has its own identity and is associated with only one set. However, identical event-command pairs, with different identities but specifying the same event and operation, could appear in different sets.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-430

Multiple Event-Command Set Specification

The Execution Platform shall permit the concurrent specification of multiple sets of event-command pairs.

Rationale: To be able to usefully associate sets of event-command pairs with operational conditions it is typically necessary to be able to specify multiple event-command sets.

Comment: A system will typically have multiple system states or modes of operation. It can therefore be useful to have multiple sets of event-command pairs, to permit each set to be associated with a given state or mode.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-440

Event-Command Control

The Execution Platform shall permit individual event-command pairs to be enabled and disabled.

Rationale: As each individual event-command pair may reflect a standalone piece of automated functionality, control over individual event-command pairs is useful.

Comment: It can be useful to be able to enable/disable individual event-command pairs to reflect changes in operational conditions.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-450

Event-Command Set Control

The Execution Platform shall permit event-command pair sets to be enabled and disabled.

Rationale: As each an event-command pair set may reflect a related set of functionality, control over whole event-command sets is useful.

Comment: It can be useful to be able to enable/disable complete event-command sets to reflect changes in system state or operational mode.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-460

Event-Command Removal

The Execution Platform shall support the removal of individual event-command pairs.

Rationale: As each individual event-command pair may reflect a standalone piece of automated functionality, removal of individual event-command pairs is useful.

Comment: Storage space for event-command pairs is finite. It is therefore likely to be important to be able to remove individual event-command pairs which are no longer operationally useful.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-470

Event-Command Set Removal

The Execution Platform shall support the removal of event-command pair sets.

Rationale: As each an event-command pair set may reflect a related set of functionality, removal of whole event-command sets is useful.

Comment: Storage space for event-command pairs is finite. It is therefore likely to be important to be able to remove event-command pairs which are no longer operationally useful. As the event-command pairs which comprise a set are likely to be functionally related, it can be useful to be able to remove the complete event-operations set, rather than requiring the removal of individual pairs.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-480

Event-Command Execution

The Execution Platform shall support the initiation of a command specified in an event-command pair when the associated event it received, provided that both the event-command pair and the set it is contained within are both enabled.

Rationale: Commands in specified as part of an event-command pair should be initiated on the emission of the specified event.

Comment: This requirement states that operations should always be initiated in response to the event specified by the pair. It makes no specific exception in relation to the previous execution of commands in response to events, which may be relevant to an implementation. It may therefore be necessary to tailor this requirement depending on the mission requirements.

Applicability: ALL
Verification Method: T

5.2.4.4 Onboard Control Procedures

Onboard control procedures (OBCPs) are software operations which are interpreted at run-time by an OBCP engine [RD.16]. The operations permitted by an OBCP are typically richer than simple sequences of operations and usually include the capability for conditional execution and loops/repetition. These requirements define the basic capabilities of OBCP execution within the context of the Execution Platform. They do not define the capabilities

of OBCP engines, or the expressive capabilities of OBCPs. These topics are considered to be outside the scope of this specification.

OSRA-EP-AUT-FN-510

OBCP Engine Provision

The Execution Platform shall provide one or more OBCP engines each of which is capable of executing OBCPs.

Rationale: An OBCP engine is necessary for the execution of OBCPs.

Comment: An OBCP engine typically provides a sandboxed environment for the execution of OBCPs. Multiple OBCP engines could be useful in order to provide isolated environments for the execution of different OBCPs.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-520

OBCP Load

The Execution Platform shall permit OBCPs to be loaded into an OBCP engine.

Rationale: To be able to use an OBCP engine for the execution of different OBCPs it must be possible to load OBCPs into the engine.

Comment: It is not required that an OBCP engine support multiple OBCPs being loaded at the same time. The initial state of the OBCP engine, e.g. whether a specific OBCP is already loaded into the engine at initialisation time, is also not specified.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-530

OBCP Deletion

The Execution Platform shall permit to delete OBCPs loaded into an OBCP engine.

Rationale: This requirement concerns the deletion of an OBCP as a resource known and loaded by the OBCP engine. It does not concern the storage location on board of the OBCP.

Comment: This requirement does not specify the behaviour of the OBCP engine if no OBCP is loaded into it.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-540

OBCP Execution Start

The Execution Platform shall permit each OBCP engine to begin executing a loaded OBCP in response to a command.

Rationale: OBCP execution should be initiated in response to a command.

Comment: The command initiating OBCP execution may have originated in an external system, or it may have originated within the Execution Platform such as, for example, from an absolute time schedule or event-command pair.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-550

OBCP Execution Stop

The Execution Platform shall permit each OBCP engine to cleanly stop executing an OBCP which is being executed in response to a command.

Rationale: It may be necessary to stop OBCP execution before completion of the OBCP if operational conditions change.

Comment: The meaning of stopping "cleanly" will depend upon the implementation but is intended to refer to a stop which, for example, may complete the execution of the current step within the OBCP. It is intended to leave the OBCP engine in a state which would permit further OBCP execution without specific intervention to reset the engine state.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-560

OBCP Execution Abort

The Execution Platform shall permit each OBCP engine to immediately abort execution of an OBCP which is being executed in response to a command.

Rationale: It may be necessary to immediately abort OBCP execution before completion of the OBCP if operational conditions change.

Comment: An "abort" is intended in contrast to a "stop" as the OBCP engine is expected to cease execution of an OBCP as quickly as possible. The resulting state of the OBCP engine may not permit further OBCP execution without specific intervention to reset the engine state.

Applicability: ALL

Verification Method: T

OSRA-EP-AUT-FN-580

OBCP Execution Status

The Execution Platform shall permit the execution status of an OBCP by an OBCP engine to be observed and monitored by other parts of the Execution Platform and by external systems.

Rationale: Automated operations should always be observable to ensure that the state of the system can be determined.

Comment: The execution status of the OBCP engine is expected to include whether an OBCP is executing (and which OBCP, if relevant) together with any kind of final result or final status information, if the OBCP has terminated.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-590

OBCP Engine Stop

The Execution Platform shall permit each OBCP engine to be stopped in response to a command. At reception of a valid OBCP Engine Stop command, the execution of any currently executing OBCP shall be aborted; all OBCPs shall be unloaded and the OBCP Engine status shall be set to “not running’.

Rationale: It may be necessary to abort all OBCPs and stop the OBCP engine, in particular in case of severe contingencies.

Comment: The OBCP engine is expected to abort executing the current OBCP as quickly as possible.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-600

OBCP Engine Execution Start

The Execution Platform shall permit each OBCP engine to start in response to a command. At reception of a valid OBCP Engine Execution Start command, the OBCP Engine status shall be set to “running”.

Rationale: Activation of OBCP Engine, which permits to perform services related to OBCPs (e.g. load, execution).

Comment: The OBCP engine is considered in a nominal status when set to “running”.

Applicability: ALL
Verification Method: T

OSRA-EP-AUT-FN-610

OBCP Execution Control

The Execution Platform shall permit the ability to invoke specific OBCPs to be enabled and disabled such that a request to begin execution of a disabled OBCP will result in an error.

Rationale: If operational conditions change, it may be useful to be able to disable a specific OBCP to ensure that it is not executed.

Comment: If an OBCP engine can only support a single loaded OBCP at any given time then disabling an OBCP is equivalent to disabling the engine. This requirement specifies that disabling an OBCP will prevent its execution from being initiated. The impact of disabling an OBCP which is currently executing (whether or not it is suspended) is not defined.

Applicability: ALL
Verification Method: T

5.2.5 Protocol Handling

Protocol Handling is identified here as a distinct capability of the Execution Platform as the majority of functions identified in the rest of this document are protocol-independent. Requirements here relate to the use of protocol in communication with external systems for the purposes of monitoring and control, and for transfer of data, such as files. An *external system* is assumed to be any other system, other than onboard avionics or mass storage devices, outside of the Execution Platform. Examples of *external systems* are operations systems on the ground, additional onboard computers and other space systems (satellites, landers, rovers, orbiters etc.).

Some degree of control over protocol handling may be provided to applications in order to permit the routing or direction of information to/from external systems according to the high-level system mode or state.

5.2.5.1 Monitoring and Control Protocol Handling

Monitoring and Control communications are likely to make up the majority of interactions between the Execution Platform and external systems, such as communications with ground.

OSRA-EP-PRT-FN-010

M&C Communication

The Execution Platform shall be capable of communicating with external systems for the purpose of achieving Monitoring and Control.

Rationale: Monitoring and Control functions need to be accessible to external systems.

Comment: This requirement specifies that there should be a protocol capable of concretely communicating the necessary interactions for the various Monitoring and Control functions as specified by Section 5.2.3.

Applicability: ALL
Verification Method: T

OSRA-EP-PRT-FN-020

M&C Communication non-functional requirements

The Execution Platform shall fulfil the applicable mission requirements for Monitoring and Control communications with external systems:

- throughput;
- reliability;
- timeliness (jitter and latency);
- coherency (in-order delivery).

Rationale: The Monitoring and Control communications mechanism must be capable of fulfilling the required mission-specific non-functional requirements.

Comment: The non-functional requirements required of Monitoring and Control communication may be mission-specific.

Applicability: ALL
Verification Method: T

OSRA-EP-PRT-FN-030

M&C Communication Tracking

The Execution Platform shall be capable of tracking and indicating to an External System the delivery, result and status of any Monitoring and Control communication received from that external system.

Rationale: The progress and integrity of the Monitoring and Control communications mechanism must be observable.

Comment: The implementation of this mechanism within a Monitoring and Control protocol is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-PRT-FN-040

M&C Communication Progress

When required by the underlying operation, the Execution Platform shall be capable of communicating the progress of an onboard operation to an external system.

Rationale: The invocation of commands supports the reporting of progress information. The Monitoring and Control protocol must similarly support the communication of this information an external system when the command is being invoked in response to Monitoring and Control communication.

Comment: The implementation of this mechanism within a Monitoring and Control protocol is not defined.

Applicability: ALL
Verification Method: T

OSRA-EP-PRT-FN-050

M&C Communication Filtering

The Execution Platform shall be capable of filtering Monitoring and Control communications with external systems such that individual types of Monitoring and Control communications can be enabled/disabled.

Rationale: Not all Monitoring and Control communications will always be applicable to all external systems under all mission conditions. By providing for the filtering of Monitoring and Control communications, the Execution Platform can be used to adapt to a range of mission applications and conditions.

Comment: Such filtering may, for example, be conducted on the basis of types of packets being exchanged, but the implementation of this mechanism within a Monitoring and Control protocol is not defined.

Applicability: ALL
Verification Method: T

5.2.5.2 Data Transfer Protocol Handling

In addition to basic Monitoring and Control, it is typically necessary for an onboard software system to be capable of delivering, or receiving, substantial data products. With “Data product” it is to be intended any data artefact, irrespectively of its format, semantics or representation (e.g., event parameters, datasets, files).

This may be direct, where the Execution Platform itself is responsible for data delivery/receipt, or indirectly, where the Execution Platform must coordinate the behaviour of an external onboard system (such as a mass memory) to achieve data delivery/receipt. In many cases the data transfer must be reliable, but the required level of

non-functional requirements (throughput, reliability, timeliness, coeherency) is deliberately left open here.

OSRA-EP-PRT-FN-110

Data Product Delivery Non-Functional Requirements

The Execution Platform shall deliver onboard data products to external systems, either directly or indirectly, by fulfilling the applicable mission requirements related to:

- throughput;
- reliability;
- timeliness (jitter and latency);
- coherency (in-order delivery).

Rationale: The data product delivery mechanism shall fulfil the non-functional requirements applicable to the mission.

Comment: The non-functional requirements of data product delivery are mission-specific.

Applicability: ALL
Verification Method: T

OSRA-EP-PRT-FN-120

Data Product Receipt

The Execution Platform shall receive onboard data products from external systems, either directly or indirectly, by fulfilling the applicable mission requirements related to:

- throughput;
- reliability;
- timeliness (jitter and latency);
- coherency (in-order delivery).

Rationale: The data product receipt mechanism shall fulfil the non-functional requirements applicable to the mission.

Comment: The non-functional requirements of data product receipt may be mission-specific.

Applicability: ALL
Verification Method: T

OSRA-EP-PRT-FN-130

Data Product Management

The Execution Platform shall provide access to onboard storage management functions (as specified in Sections 5.2.2.3 and 5.2.2.4) to external systems as these relate to the management of data products.

Rationale: The communications mechanism used for data product delivery and/or receipt must be capable of providing facilities for the remote management of stored data products.

Comment: These functions may be file management and/or packet store management depending on the chosen mechanism for data product representation.

Applicability: ALL

Verification Method: T

5.2.5.3 Onboard Communications Protocol Handling

Where a spacecraft comprises multiple computing platforms, communications between computers is necessary, which requires the handling of suitable protocols. For example, in the case of a platform and payload computer, communications between the two computers requires suitable protocol handling.

OSRA-EP-PRT-FN-210

Multi-node Onboard Communication

The Execution Platform shall be capable of communicating with other onboard computers, where required. It is left to mission-specific requirements whether those computers belong to the same avionics and software system, or shall be considered external systems.

Rationale: Where a spacecraft has multiple onboard computers, communications between these computers is likely to be necessary.

Comment: Note that communications between computers, or other devices running software which falls within the scope of the OSRA, is distinct from communications with onboard avionics devices.

Applicability: ALL

Verification Method: T

5.2.6 On-board Communication and Device Access

The On-board Communication and Device Access capability relates specifically to the ability to interact with onboard devices, using onboard communication systems. It is divided into two aspects: the ability to carry out basic operations over an on-board communication systems; and the ability to interact with devices connected to an onboard communication system. Additionally, the Execution Platform may provide a higher-level, “idealised”, interface to the device which corresponds to some form of accepted standard

behaviour for the device type with telemetry provided in standard engineering units (such as SI units)

As was discussed in Section 4, although these requirements identify distinct functions of the Execution Platform, it is permitted to not implement them as distinct or separable blocks or modules.

5.2.6.1 On-board Communication Access

To be able to interact with onboard devices, the Execution Platform must provide the necessary facilities to use and manage the connection(s) to those devices via onboard communication systems.

OSRA-EP-ODA-FN-010

On-board Communication Access

The Execution Platform shall provide access to each onboard communication systems, with the operations supported reflecting the semantics of the underlying technology.

Rationale: The ability to access on-board communication systems, and the devices connected to them, is considered to be part of the Execution Platform.

Comment: The operations supported for accessing an on-board communication system are expected to be defined by the characteristics of the specific data communication technology. For example, some onboard communication systems may conduct communications in terms of "reads" or "writes" from a specific initiator to a target. In other cases, on-board communication operations may be conducted in terms of the exchange of packets asynchronously between peers.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-020

On-board Communication Operation Provision

The Execution Platform shall provide at least those operations necessary to support the devices present on a given on-board communication system.

Rationale: Whilst an on-board communication technology (e.g., Spacewire, SpaceFibre, 1553) may provide for a wide range of operations, it is only necessary for an Execution Platform to support those operations needed to communicate with the devices effectively present on the on-board communication system for the given mission.

Comment: The capabilities of some on-board communication technologies are extensive. The intention of this requirement is to make it clear that generic support for all possible operations on a given technology is not expected if not required for the specific avionics or mission requirements.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-030

On-board Communication Operation Synchronisation

The Execution Platform shall provide the capability to synchronise operations on an on-board communication system to either a software-derived timing source, an on-board communication system derived timing source or an independent hardware-derived timing source where such a capability is necessary to support the device access patterns required for the mission.

Rationale: To ensure a deterministic pattern of operations on an on-board communication system, synchronisation to a timing source is frequently necessary.

Comment: This only requires that the Execution Platform provide for synchronisation of on-board communication operations where this is required for the mission.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-040

On-board Communication Management

The Execution Platform shall provide the capability to configure and manage on-board communication system resources, such as link/bus interfaces and routing switches, where such resources are present.

Rationale: The management of on-board communication system resources (when present) falls within the scope of the Execution Platform.

Comment: It is not possible for applications to manage on-board communication resources as these relate to communications with the attached on-board communication devices rather than the high-level behaviour of the attached devices themselves. On-board communication link/bus interfaces and routing switches are therefore considered to be part of the on-board communication system.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-050

On-board Communication Discovery

The Execution Platform shall provide the capability to interrogate and/or discover the topology of an on-board communication system, together with the basic identities of devices present on it, should the underlying technology support this.

Rationale: Some on-board communication system technologies provide the capability to discover the topology and/or attached devices. This functionality can be used to detect device presence and to identify device and on-board communication system failures.

Comment: This requirement has been included to reflect the capabilities of some technologies together with their anticipated use. It is expected that this requirement will frequently be removed through tailoring, as it may not be applicable to all missions.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-060

On-board Communication System Redundancy Management

The Execution Platform shall provide the capability to manage on-board communication system redundancy at both the link and network/bus levels.

Rationale: The management of on-board communication system resources (when present) falls within the scope of the Execution Platform. This requirement does not however implies that the management of redundancy is implemented by the Execution Platform, only the mechanisms to act upon it.

Comment: Whilst the management of device redundancy is expected to be possible at application level, the management of on-board communication redundancy must be conducted within the Execution Platform. This may be done in response to information from applications.

Applicability: ALL
Verification Method: T

5.2.6.2 Device Commanding and Data Acquisition

Building on the facilities provided to access the underlying on-board communication systems, the Execution Platform is also responsible for providing access to avionics devices, and to permit their management.

OSRA-EP-ODA-FN-110

Device Access

Making use of On-board Communication Access functions, the Execution Platform shall provide the ability to communicate with all relevant avionics devices on board.

Rationale: The communication with onboard avionics devices via onboard communication systems falls within the scope of the Execution Platform.

Comment: It is not desirable for applications to conduct low-level communications with avionics devices as this relates to communications with the device rather than the high-level behaviour of the devices. It is intended that applications should only consider device behaviour rather than communications.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-120

Low-Level Device Interaction

The Execution Platform shall permit interaction with avionics devices in terms of the basic operations defined by each device, including the ability to command and configure devices and sample device telemetry.

Rationale: The Execution Platform shall have access to all necessary functionality of a device.

Comment: Whilst a device may be made accessible to applications in terms of its high-level behaviour, potentially through some form of abstraction, the Execution Platform should be capable of interacting with the device at a low level.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-130

High-Level Device Commanding

The Execution Platform shall provide the capability to command and configure the avionics device in terms of an accepted set of standard behaviours for the relevant type of avionics device.

Rationale: The low-level characteristics and behaviour of a device may vary significantly between implementations, even within a specific device type (such as a gyroscope or reaction wheel). To permit a greater level of functional reuse, the implementation-specific low-level behaviour can be abstracted to create a standard high-level behaviour.

Comment: An accepted standard for the high-level behaviour of a device may not exist. Even where it does, it may be decided that such an abstraction is unnecessary, in which case it is expected that this requirement would be removed through tailoring. SAVOIR-SAFI [RD.17] has defined an initial high-level common set of commands and telemetry for a number of on-board devices

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-140

High-Level Device Acquisition

The Execution Platform shall provide the capability to convert telemetry values sampled from avionics devices from raw, measured quantities to standard engineering units.

Rationale: The representation of telemetry quantities from a device may vary significantly between implementations, even within a specific device type (such as a gyroscope or reaction wheel). To permit a greater level of functional reuse, the implementation-specific raw values can be converted into standard engineering units.

Comment: An accepted standard engineering unit for a given telemetry value may not exist. Even where it does, it may be decided that such a conversion is unnecessary on board, in which case it is expected that this requirement would be removed through tailoring.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-150

Device Access Synchronisation

The Execution Platform shall provide the capability to synchronise avionics device operations to either a software-derived timing source, an on-board communication system-derived timing source or an independent hardware-derived timing source where such a capability is necessary to support the required device access patterns.

Rationale: To ensure a deterministic pattern of operations to access an avionics device, synchronisation to a timing source may be necessary. It is expected that synchronisation facilities will be used to achieve device access synchronisation.

Comment: This only requires that the Execution Platform provide for synchronisation of device access operations where this is required for the mission.

Applicability: ALL
Verification Method: T

OSRA-EP-ODA-FN-160

Device Redundancy Management

The Execution Platform shall provide the capability to manage avionic device redundancy where relevant. If required, this should include the potential to manage either hot or cold (or warm) redundancy.

Rationale: Although applications may direct the use of device redundancy, the implementation of redundancy management of devices falls within the scope of the Execution Platform.

Comment: Decisions as to how and when redundant devices should be used may be taken either within the Execution Platform or by applications. Where these decisions are taken by applications, it is up to the Execution Platform to manage the necessary operations to achieve the redundancy switching/failover.

Applicability: ALL
Verification Method: T

5.2.7 Access by External Systems

As the Execution Platform is expected to be used onboard a spacecraft, the platform is likely to be physically inaccessible to external systems which makes system maintenance and issue diagnosis difficult. To help alleviate these issues, the Execution Platform provides facilities for accessing onboard storage at a low level, as well as low-level access to avionics devices.

5.2.7.1 Remote Platform Access

To facilitate platform maintenance, the Execution Platform provides low-level access to onboard storage, including storage for the software image and the working memory being used for software execution. Such access must clearly be used with great care but low-level access is an important provision in ensuring that unforeseen scenarios can be appropriately recovered.

OSRA-EP-AES-FN-010

Memory Read Access by External Systems

The Execution Platform shall provide the capability for external systems to read from all externally-accessible onboard memory and storage devices at the byte level.

Rationale: The diagnosis of some onboard issues may require low-level access to onboard memory and storage. Yet, access to some memory zone can be restricted for security reason (e.g. memory used to store encryption keys).

Comment: As with Monitoring and Control requirements, this requirement specifies the existence of a capability as distinct from the communications protocol used to access it.

Applicability: ALL

Verification Method: T

OSRA-EP-AES-FN-020

Memory Write Access by External Systems

The Execution Platform shall provide the capability for external systems to write to all externally-accessible onboard memory and storage devices at the byte level where such memory/storage is writable.

Rationale: The diagnosis of some onboard issues may require low-level access to onboard memory and storage. Additionally, write access may be used for some software maintenance tasks, such as updating software images and applying software patches. Yet, access to some memory zone can be restricted for security reason (e.g. memory used to store encryption keys).

Comment: As with Monitoring and Control requirements, this requirement specifies the existence of a capability as distinct from the communications protocol used to access it.

Applicability: ALL

Verification Method: T

5.2.7.2 Remote Device Access

Diagnosing issues with a remote system using the Execution Platform may require interaction with avionics devices in ways which were not originally foreseen as part of nominal operations. It is therefore important that the Execution Platform support low-level access to avionics devices such that important on-board communication interactions can be invoked and/or controlled by external systems.

OSRA-EP-AES-FN-110

On-board Communication Access by External Systems

The Execution Platform shall provide the capability for external systems to carry out operations of onboard communication systems, with the operations supported reflecting the semantics of the underlying technology.

Rationale: The diagnosis of some onboard issues may require low-level access to on-board communication systems.

Comment: It is expected that the Execution Platform will make available to external systems the full range of on-board communication access functions defined in Section 5.2.6.1.

Applicability: ALL

Verification Method: T

OSRA-EP-AES-FN-120

Device Access by External Systems

The Execution Platform shall provide the capability for external systems to interact with avionics devices in terms of the basic operations defined by each device, including the ability to command and configure devices and sample device telemetry.

Rationale: The diagnosis of some onboard issues may require low-level access to devices.

Comment: It is expected that the Execution Platform will make available to external systems the low-level device access functions defined in Section 5.2.6.2.

Applicability: ALL

Verification Method: T

5.2.7.3 Time Access and Correlation

Monitoring and controlling a remote system frequently requires a consistent understanding of time such that remote system time, i.e. onboard time, can be accurately correlated with an external time source, such as ground time. The Execution Platform must therefore provide facilities to assist in time correlation with external systems.

OSRA-EP-AES-FN-210

Time Access by External Systems

The Execution Platform shall provide the capability for external systems to query the value of onboard time sources.

Rationale: The correlation of external time sources with time sources onboard requires access to the onboard time.

Comment: It is expected that the Execution Platform will make available to external systems the value of all onboard time sources as defined in Section 4.2.2.2.

Applicability: ALL

Verification Method: T

OSRA-EP-AES-FN-220

Time Synchronisation with External Systems

The Execution Platform shall provide the capability for the reply to a time query request by an external system to be synchronised to the RF communications system such that the onboard latency and jitter of the time query reply are characterised.

Rationale: The precise correlation of external time sources with onboard time requires that the timing of a reply to a request for an onboard time from an external system is characterised and known.

Comment: This requirement does not define the means by which the onboard time value is returned or how the timing of the reply should be characterised.

Applicability: ALL
Verification Method: T

5.3 System Interface Requirements

It is expected that only a subset of Execution Platform functions will be accessible to applications. Many functions exist entirely within the Execution Platform whilst others may only be accessible to external systems and not to applications. The required functional interface to applications is therefore relatively small, and is specified by the requirements in this section.

The functional breakdown in this section is chosen to correspond to the interface expectations of higher layers in the OSRA (see [RD.3]).

5.3.1 General Capabilities

OSRA-EP-GEN-IF-010

State Observability

The Execution Platform shall ensure that any state which may be modified by applications, such as the ability enable or disable a function, is also observable to applications.

Rationale: It is important that applications can confirm or determine the state of any Execution Platform function that they may affect directly to ensure proper coordination between application and Execution Platform behaviour.

Comment: Applications can use the Execution Platform interface to directly affect the state and behaviour of the Execution Platform. It is important that this state is observable as, otherwise, it might be necessary to make attempts to track Execution Platform state within applications, leading to the possibility of this information becoming invalid.

Applicability: ALL
Verification Method: RoD, T

OSRA-EP-GEN-IF-020

Tasking and Concurrency Interface

The Execution Platform shall make available to applications all tasking and concurrency functions as specified in Section 5.2.1.3.

Rationale: Tasking and concurrency functions are typically necessary for the correct function of applications.

Comment: Tasking and concurrency functions are as necessary for applications as they are for the Execution Platform itself. It is therefore important that the Execution Platform interfaces exposes this functionality to applications.

Applicability: ALL
Verification Method: RoD. T

OSRA-EP-GEN-IF-030

Safeguard Memory Interface

The Execution Platform shall make available to applications the ability to read and write to application context memory stored in the primary and secondary safeguard regions as specified in Section 5.2.1.5.

Rationale: Safeguard memory is expected to be used by applications for context storage. It is therefore important that the Execution Platform interface provides both read and write access to regions within safeguard memory.

Comment: The way in which safeguard memory is used to provide persistent context storage for applications is Execution Platform and application-specific. This requirement specifies that the necessary functionality be made available, not the form that should take.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-040

Libraries Interface

The Execution Platform shall make available to applications all relevant software libraries as specified in Section 5.2.1.6.

Rationale: Although support libraries are considered to be part of the Execution Platform, they may only be present in the system for the use of applications. It is therefore important that an interface to the libraries is provided to applications.

Comment: It is expected that access to libraries will be provided by the Execution Platform "transparently" such that the defined interface to the library functionality is made available, in full, to applications as a matter of course.

Applicability: ALL
Verification Method: RoD

5.3.2 Reporting

OSRA-EP-GEN-IF-110

Parameter Report Generation Control Interface

The Execution Platform shall make available to applications the ability to enable and disable parameter report generation as specified in Section 5.2.3.2.

Rationale: Different sets of parameter reports are frequently related to different system states or operational modes. It is expected that application software will determine the current state/mode and therefore should have control over which parameter reports are generated.

Comment: It is expected that specific parameter reports will be associated with particular system states or operational modes as part of the application software design, and that the Execution Platform will be configured accordingly. The application software may not be able to influence the composition of the parameter report, or be able to add/remove parameter reports, but it can use knowledge of the existence of parameter reports to control their generation.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-120

Event Forwarding Control Interface

The Execution Platform shall make available to applications the ability to enable and disable event forwarding to external systems as specified in Section 5.2.3.5.

Rationale: Different sets of events are often more applicable to some system states or operational modes than others. It is expected that application software will determine the current state/mode and therefore should have control over which events are forwarded to external systems.

Comment: It is expected that specific events will be more relevant in particular system states or operational modes and this will be known during the application software design permitting the Execution Platform to be configured accordingly. The application software may not be aware of the precise identities of the events being enabled/disabled (as they may form part of a set), instead the identification being used by applications is related to the purpose of the events. Whilst the focus of this interface is control over forwarding to external systems, it may also be used by the Execution Platform to control other functionality, such as logging.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-130

Dataset Forwarding Control Interface

The Execution Platform shall make available to applications the ability to enable and disable dataset forwarding to external systems as specified in Section 5.2.3.6.

Rationale: Different sets of datasets are often more applicable to some system states or operational modes than others. It is expected that application software will determine the current state/mode and therefore should have control over which datasets are forwarded to external systems.

Comment: It is expected that specific datasets will be more relevant in particular system states or operational modes and this will be known during the application software design permitting the Execution Platform to be configured accordingly. The application software may not be aware of the precise identities of the datasets being enabled/disabled (as they may form part of a set), instead the identification being used by applications is related to the purpose of the datasets. Whilst the focus of this interface is control over forwarding to external systems, it may also be used by the Execution Platform to control other functionality, such as logging.

Applicability: ALL
Verification Method: T

5.3.3 Monitoring

OSRA-EP-GEN-IF-210

Monitoring Control Interface

The Execution Platform shall make available to applications the ability to enable and disable monitoring checks as specified in Section 5.2.3.4.

Rationale: Different sets of monitoring checks are frequently related to different system states or operational modes. It is expected that application software will determine the current state/mode and therefore should have control over which monitoring checks are enabled.

Comment: It is expected that specific monitoring checks will be associated with particular system states or operational modes as part of the application software design, and that the Execution Platform will be configured accordingly. The application software may not be able to influence the configuration of the monitoring check (such as its type or limits), or be able to add/remove monitoring checks, but it can use knowledge of the existence of monitoring checks to control their use through the ability to enable/disable them.

Applicability: ALL
Verification Method: T

5.3.4 Commanding

OSRA-EP-GEN-IF-310

Commanding Interface

The Execution Platform shall make available the ability invoke commands as specified in Section 5.2.3.1.

Rationale: The capability of invoking commands of the Execution platform or of applications is provided by the Execution Platform

Applicability: ALL
Verification Method: T

5.3.5 Automation

OSRA-EP-GEN-IF-410

Absolute and Relative Time Automation Control Interface

The Execution Platform shall make available to applications the ability to enable and disable absolute and relative time operation sequences as specified in Section 5.2.4.1.

Rationale: Different time-based automation schedules/sequences are frequently related to different system states or operational modes. It is expected that application software will determine the current state/mode and therefore should have control over which schedules/sequences are enabled.

Comment: It is expected that specific absolute time schedules or relative time operational sequences will be associated with particular system states or operational modes as part of the application software design, and that the Execution Platform will be configured accordingly. The application software may not be able to influence the configuration of schedules or sequences, or be able to add/remove schedules/sequences, but it can use knowledge of the existence of these to control their use through the ability to enable/disable them.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-420

Position Automation Control Interface

The Execution Platform shall make available to applications the ability to enable and disable position operation sequences as specified in Section 5.2.4.2.

Rationale: Different position-based automation sequences are frequently related to different system states or operational modes. It is expected that application software will determine the current state/mode and therefore should have control over which sequences are enabled.

Comment: It is expected that specific position operation sequences will be associated with particular system states or operational modes as part of the application software design, and that the Execution Platform will be configured accordingly. The application software may not be able to influence the configuration of position sequences, or be able to add/remove sequences, but it can use knowledge of the existence of sequences to control their use through the ability to enable/disable them.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-430

Event Automation Control Interface

The Execution Platform shall make available to applications the ability to enable and disable event-command pairs and sets as specified in Section 5.2.4.3.

Rationale: Different event-based automation pairs (event-command associations) are frequently related to different system states or operational modes. It is expected that application software will determine the current state/mode and therefore should have control over which event-command pairs, or sets of pairs, are enabled.

Comment: It is expected that specific event-command pairs will be associated with particular system states or operational modes as part of the application software design, and that the Execution Platform will be configured accordingly. The application software may not be able to influence the configuration of specific pairs (by, for example modifying the event associated with a specific operation), or be able to add/remove pairs, but it can use knowledge of the existence of pairs or sets of pairs to control their use through the ability to enable/disable them.

Applicability: ALL
Verification Method: T

5.3.6 Forwarding

OSRA-EP-GEN-IF-510

Forwarding Control Interface

The Execution Platform shall make available to applications the ability to enable and disable types of Monitoring and Control communication with external systems as specified in Section 5.2.5.1.

Rationale: The use of links to external systems, such as a space link, is frequently related to different system states or operational modes. It is expected that application software will determine the current state/mode and therefore should have control over the use of links to external systems.

Comment: Whilst other parts of the Execution Platform interface permit control of the generation of Monitoring and Control information, this requirement specifically relates to control of the communication of such information to external systems, and therefore to the bandwidth on external system links. It is expected that communications at the link level will be identified by type (e.g. packet type) and that this interface will permit control of different types of communication. The identification used by applications to specify which types of communication are to be enabled/disabled is implementation-specific but it is anticipated that an identifier would be used, with the configuration of the Execution Platform determining the communication types (e.g. packet types) that this refers to.

Applicability: ALL

Verification Method: T

5.3.7 Onboard Control Procedures

OSRA-EP-GEN-IF-610

OBCP Control Interface

The Execution Platform shall make available to applications the ability to start, stop and abort OBCPs as specified in Section 5.2.4.4.

Rationale: The existence of some OBCPs (typically Onboard Application Procedures or OBAPs) is expected to be visible to applications. As such applications may need to control the execution of these OBCPs as part of high-level mission operations.

Comment: It is expected that specific OBCPs will be associated with particular operations as part of the application software design, and that a suitable OBCP implementing that procedure will be loaded into the relevant OBCP engine. The application software may not be able to load or delete the OBCP itself, but it can use knowledge of its existence to control execution of the operation(s) it implements. It is expected that the operational purpose of the OBCP will be maintained through the life of the system, even if the implementation is modified by loading a new OBCP in its place.

Applicability: ALL
Verification Method: T

5.3.8 Platform Management

OSRA-EP-GEN-IF-710

Platform Restart Interface

The Execution Platform shall make available to applications the ability to request software and hardware platform restart as specified in Section 5.2.2.1.

Rationale: As identified in Section 5.2.2.1 it is typically necessary for application software to be able to restart the software and/or hardware platform as part of FDIR.

Comment: The requirements in Section 5.2.2.1 identify the existence of application-facing functionality for restarts of the hardware and software platform; this requirement specifies that such functionality forms part of the Execution Platform interface.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-720

Non-Fatal Error Reporting Interface

The Execution Platform shall make available to applications the ability to report non-fatal errors and receive non-fatal error indications as specified in Section 5.2.1.4.

Rationale: As identified in Section 5.2.1.4 it is often necessary for application software to be able to report and receive non-fatal error indications as part of FDIR.

Comment: The requirements in Section 5.2.1.4 identify the existence of application-facing functionality for the reporting and receiving of non-fatal error indications; this requirement specifies that such functionality forms part of the Execution Platform interface.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-730

Fatal Error Reporting Interface

The Execution Platform shall make available to applications the ability to report fatal errors as specified in Section 5.2.1.4.

Rationale: As identified in Section 5.2.2.1 it is often necessary for application software to be able to report fatal errors as part of FDIR.

Comment: The requirements in Section 5.2.2.1 identify the existence of application-facing functionality for the reporting of fatal error indications; this requirement specifies that such functionality forms part of the Execution Platform interface.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-740

Partition Management Interface

The Execution Platform shall make available to applications all partition management functions as specified in Section 5.2.1.2.

Rationale: As identified in Section 5.2.1.2 it is often necessary for parts of the application software in a TSP system to be able to control partition execution as part of FDIR.

Comment: The requirements in Section 5.2.1.2 identify the existence of application-facing functionality for partition management; this requirement specifies that such functionality forms part of the Execution Platform interface.

Applicability: TSP
Verification Method: T

5.3.9 Time Access

OSRA-EP-GEN-IF-810

Time Access Interface

The Execution Platform shall make available to applications the ability to request the current time from any onboard time source supported by the Execution Platform as specified in Section 4.2.2.2. This capability shall be restricted to applications having appropriate security privileges, if this notion is implemented on board.

Rationale: The types of functionality expected to be implemented by applications, relating typically to overall mission management and behaviour, may be time-dependent and may therefore rely on access to an onboard time source.

Comment: It is expected that applications will be permitted read access to any available onboard time source, if security privileges permit, but that write access will not in the general case be permitted.

Applicability: ALL
Verification Method: T

5.3.10 Device Access

OSRA-EP-GEN-IF-910

Low-Level Device Access Interface

The Execution Platform shall make available to applications the ability to interact with avionics devices in terms of the basic operations defined by each device, including the ability to command and configure devices and sample device telemetry as specified in Section 5.2.6.2.

Rationale: The types of functionality expected to be implemented by applications, relating typically to overall mission management and behaviour which may include (for example) attitude and orbit control, may require interaction with onboard avionics devices.

Comment: Not all devices, and not all device functionality, may be accessible through the Execution Platform interface. Access to devices by applications will need to be coordinated with device access by the Execution Platform itself. The mechanisms for such coordination are undefined.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-920

High-Level Device Access Interface

The Execution Platform shall make available to applications the ability to interact with avionics devices in terms of an accepted set of standard behaviours for the relevant type of avionics device with telemetry being supplied in standard engineering units as specified in Section 5.2.6.2.

Rationale: The low-level characteristics and behaviour of a device may vary significantly between implementations, even within a specific device type (such as a gyroscope or reaction wheel). To permit a greater level of functional reuse for applications, the implementation-specific low-level behaviour can be abstracted to create a standard high-level behaviour.

Comment: An accepted standard for the high-level behaviour of a device may not exist although it would be possible for a set of accepted high-level device behaviours to be defined for a specific Execution Platform implementation, giving the potential for application independence from specific device implementations.

Applicability: ALL
Verification Method: T

OSRA-EP-GEN-IF-930

Device Data Pool Interface

The Execution Platform shall make available avionics device telemetry which has been pooled through periodic or asynchronous acquisition as part of Monitoring and Control features as specified in Section 5.2.3.3.

Rationale: Depending on the access pattern to device telemetry used by the Execution Platform, telemetry may be acquired periodically and placed in a pool. In this case it is appropriate to give applications access to the pooled data either instead of, or as well as, direct access to device telemetry.

Comment: Providing applications with access to a pool is one way of resolving issues around the coordination of applications and the Execution Platform for telemetry requests to the device.

Applicability: ALL
Verification Method: T

5.3.11 File Access

OSRA-EP-GEN-IF-1000

File Access Interface

The Execution Platform shall make available to applications the ability to interact with files in terms of an accepted set of standard behaviours as specified in section 5.2.2.3.

Rationale: The Execution Platform comprises all file systems and management capabilities.

Comment: The ability to access and manage a file system, where present, is considered to be part of the Execution Platform, which will make it available to the application via an appropriate interface. This can be, e.g., a dedicated API, or a pseudo-component.

Applicability: ALL
Verification Method: T

6 TOPICS TO BE ADDRESSED IN NEXT RELEASE

The current version of the document implements all the dispositions to major and minor RIDs following the SAVOIR-FAIRE review of the document (Issue 1 revision 0).

The following topics are left open for resolution in the next release:

- Introduction of a notion of minimum capability set
- More detailed mapping to PUS services
- A review and better explanation of variability levels of the OBSW in the OSRA, and in particular of the Execution Platform
- A possible restructuring of placement of descriptions between head chapters and sub-chapters
- The initialisation of non-critical applications in TSP environments, as well as more detailed requirements for reboot of partitions in TSP environment.

APPENDIX A RELATION TO EXISTING STANDARDS

This Execution Platform functional specification has been designed deliberately to be agnostic to implementation standards and a range of standards were used as a reference when specifying Execution Platform functions.

A.1 ECSS Packet Utilisation Standard

This section offers an informal trace between the Execution Platform functions and the ECSS PUS standard [RD.15] including all key services.

Capability	**Function**	**PUS Feature/Service**
Software Execution Environment	*All*	Not relevant to PUS
Hardware Execution Environment	*All*	Not relevant to PUS
Monitoring and Control	Application Commanding	Service 8/Custom services
	Parameter Access and Reporting	Service 20, Service 3
	Parameter Acquisition and Pooling	Not specifically relevant to PUS
	Parameter Statistics and Monitoring	Service 4, Service 12
	Event Distribution and Reporting	Service 5
	Dataset Distribution and Reporting	Custom services
	Onboard Logging	Service 15
Automation	Time-Based Automation	Service 11, Service 21
	Position-Based Automation	Service 22
	Event-Based Automation	Service 19
	Onboard Control Procedures	Service 18

Protocol Handling	Monitoring and Control Protocol Handling	All packet encoding/decoding, Service 1, Service 14, Service 17
	Data Transfer Protocol Handling	Service 13 (and CFDP)
	Onboard Communications Protocol Handling	*May be implemented with PUS, may not be*
On-board Communication and Device Access	*All*	Not relevant to PUS
Access by External Systems	Remote Platform Access	Service 6, Service 23
	Remote Device Access	Service 2
	Time Access and Correlation	Service 9

A.2 CCSDS Mission Operations Services

This section offers an informal trace between the Execution Platform functions and the framework and standard services of Mission Operations [RD.6].

Capability	**Function**	**MO Feature/Service**
Software Execution Environment	*All*	Not relevant to MO
Hardware Execution Environment	*All*	Not relevant to MO
Monitoring and Control	Application Commanding	Action/Custom services
	Parameter Access and Reporting	Parameter, Aggregation
	Parameter Acquisition and Pooling	Not specifically relevant to MO

	Parameter Statistics and Monitoring	Statistics, Check
	Event Distribution and Reporting	Alert
	Dataset Distribution and Reporting	Custom services
	Onboard Logging	Archive
Automation	Time-Based Automation	Scheduling, Automation
	Position-Based Automation	Scheduling
	Event-Based Automation	Automation
	Onboard Control Procedures	Automation
Protocol Handling	Monitoring and Control Protocol Handling	MAL plus bindings
	Data Transfer Protocol Handling	(CFDP and potentially other protocols)
	Onboard Communications Protocol Handling	*May be implemented with MO, may not be*
On-Board Communication and Device Access	*All*	Not relevant to MO
Access by External Systems	Remote Platform Access	Software Management, File Management
	Remote Device Access	Custom services
	Time Access and Correlation	Time