Packet Utilization Services for GNURadio

Description and User Manual

Document: PUS-042104-UM-00100

RELEASE: ***A***

DATE: April 10th, 2024

Author: Gustavo Gonzalez

ARGENTINA

INDEX

CHANGE LOG		
ISSUE	**CHANGE DESCRIPTION**	**DATE**
A	Initial	April 10th, 2024

1 INTRODUCTION

The gr-pus OOT package incorporates the Packet Utilization Services ECSS-E-ST-70-41C [A.D.1] interfaces and services into GNURadio.

Additional ad-hoc (application specific) services could easily be added later as OOT blocks.

This package would required additional code in order to bind the services with each specific application, ie binding the parameters with hardware sensors.

This implementation is based in the AcubaSat one, see [R.D.1], but adapted to GNURadio and including additional service types and message types.

This document is intented to summarize the gr-pus implementation and its interfaces, for a full description of each service and subtypes specification see [A.D.1].

1.1 APPLICABLE AND REFERENCE DOCUMENTS

Applicable Documents hereinafter are referred to as AD.X and the reference ones as RD.X.

[A.D.1] ECSS-E-ST-70-41C Telemetry and telecommand packet utilization

[R.D.1] https://github.com/AcubeSAT/ecss-services
[R.D.2] https://www.gnuradio.org/

1.2 DEFINITIONS

According to [A.D.1], next definitions are used:

1. ***Parameters.*** A parameter is a value (often numerical) that represents a small piece of data which can be sent to or received from the satellite. Parameters can represent sensor outputs, configuration values, status indicators, or everything else.
   Parameters are mainly handled by the ParameterService, this these service has the capability to modify parameters values aimed to set

mission input variables settings, but is up to the specific mission code to update any other parameter value as physical variables readings (i.e.temperatures)
Parameters are used for the Housekeeping service to build telemetries frames, they are used for On board monitoring service to keep tracking of the satellite condition, also they are used by the Statistic service to keep tracking of parameters variation through time

2. ***Events***. Events represent expected or unexpected occurences on the spacecraft.
The EventReportService is mainly responsible for management of on-board events reporting while EventActionService execute actions on events detection. Other services provide the capability of generating or responding to on-board events. The events are mainly trigger by the On board monitoring service or any other service with the capability to trigger events

3. ***Application Processes (AP).*** An Application Process is any physical (hardware) or logical (software) entity that can handle PUS messages.
In most cases, an Application Process will be a single microcontroller or subsystem. For example, the OBC, ADCS and Ground Station may be different Application Processes.

2 PACKET UTILIZATION SERVICE IMPLEMENTATION

Next matrix resumes the PUS services implementation fulfilling the minimum tailoring required by [A.D.1]:

Service		implemented	Unitary Tested
[ST01]	Request Verification	implemented	Tested
[ST02]	Device Access	No implemented in this version	
[ST03]	Housekeeping	implemented	Tested
[ST04]	Parameter Statistics	implemented	Tested
[ST05]	Event Report	implemented	Tested
[ST06]	Memory Management	implemented (*)	Tested
[ST08]	Function Management	implemented	Tested
[ST09]	Time Management	not implementation planned	
[ST11]	Time Based Scheduling	implemented	Tested
[ST12]	On Board Monitoring	implemented	Tested
[ST13]	Large Packet Transfer	implemented	Tested

[ST14]	Real Time Forwarding Control	implemented	Tested
[ST15]	Storage And Retrieval	implemented	Tested
ST[17]	Test	implemented	Tested
ST[18]	On-board Operations Procedure	not implementation planned	
[ST19]	Event Action	implemented	Tested
[ST20]	Parameter Management	implemented	Tested
ST[21]	Request Sequencing	implemented	Tested
ST[22]	Position-based Scheduling	No implemented in this version	
ST[23]	File Management	implemented	Tested

(*) hardware dependence singleton required to interface memory with service, a test singleton has been used for unitary testing

Service		interface	message type	required capability	implemented capability
[ST01] Request Verification	Acceptance and reporting	TM[1,1]	successful acceptance verification report	minimum	implemented
		TM[1,2]	failed acceptance verification report	minimum	implemented
	Execution reporting	TM[1,3]	successful start of execution verification report	minimum	implemented
		TM[1,4]	failed start of execution verification report	minimum	implemented
		TM[1,5]	successful progress of execution verification report	minimum	implemented
		TM[1,6]	failed progress of execution verification report	minimum	implemented
		TM[1,7]	successful completion of execution report	minimum	implemented
		TM[1,8]	failed completion of execution verification report	minimum	implemented
	Routing and reporting	TM[1,10]	failed routing verification report	minimum	implemented
Service		interface	message type	required capability	implemented capability
[ST02] Device Access		TC[2,1]	distribute on/off device commands	minimum, at least one of: TC[2,1], TC[2,2], TC[2,4], TC[2,7]	
		TC[2,2]	distribute register load commands	minimum, at least one of: TC[2,1], TC[2,2], TC[2,4], TC[2,7]	
		TC[2,5]	distribute register dump commands	requires TC[2,2]	
		TM[2,6]	register dump report	TC[2,5] response	
		TC[2,4]	distribute CPDU commands	minimum, at least one of: TC[2,1], TC[2,2], TC[2,4], TC[2,7]	

		TC[2,7]	distribute physical device commands		minimum, at least one of: TC[2,1], TC[2,2], TC[2,4], TC[2,7]	
		TC[2,8]		acquire data from physical devices	implied by TC[2,7]	
		TM[2,9]		physical device data report	TC[2,8] response	
		TC[2,10]		distribute logical device commands	requires TC[2,7]	
		TC[2,11]		acquire data from logical devices	implied by TC[2,10]	
		TM[2,12]		logical device data report	TC[2,11] response	
Service		interface	message type		required capability	implemented capability
ST[03] Housekeeping	Housekeeping reporting	TM[3,25]	housekeeping parameter report		minimum TC[3,27] response	implemented
		TC[3,5]	enable the periodic generation of housekeeping parameter reports		by declaration	implemented
		TC[3,6]		disable the periodic generation of housekeeping parameter reports	implied by TC[3,5]	implemented
		TC[3,1]	create a housekeeping parameter report structure		by declaration	implemented
		TC[3,3]		delete housekeeping parameter report structures	implied by TC[3,1]	implemented
		TC[3,9]		report housekeeping parameter report structures	requires TC[3,1]	implemented
		TM[3,10]		housekeeping parameter report structure report	TC[3,9] response	implemented
		TC[3,29]		append parameters to a housekeeping parameter report structure	requires TC[3,1]	implemented
		TC[3,31]	modify the collection interval of housekeeping parameter report structures		by declaration	implemented

		TC[3,33]	report the periodic generation properties of housekeeping parameter report structures	by declaration	implemented
		TM[3,35]	housekeeping parameter report periodic generation properties report	TC[3,33] response	implemented
		TC[3,27]	generate a one shot report for housekeeping parameter report structures	by declaration	implemented
	Diagnostic reporting	TM[3,26]	diagnostic parameter report	minimum	
		TC[3,7]	enable the periodic generation of diagnostic parameter reports	minimum	
		TC[3,8]	disable the periodic generation of diagnostic parameter reports	minimum	
		TC[3,2]	create a diagnostic parameter report structure	minimum	
		TC[3,4]	delete diagnostic parameter report structures	minimum	
		TC[3,11]	report diagnostic parameter report structures	requires TC[3,2]	
		TM[3,12]	diagnostic parameter report structure report	TC[3,11] response	
		TC[3,30]	append parameters to a diagnostic parameter report structure	requires TC[3,2]	
		TC[3,32]	modify the collection interval of diagnostic parameter report structures	by declaration	
		TC[3,34]	report the periodic generation properties of diagnostic parameter report structures	by declaration	
		TM[3,36]	diagnostic parameter report periodic generation properties report	TC[3,34]	
		TC[3,28]	generate a one shot report for diagnostic parameter report structures	by declaration	
		TM[3,26]	diagnostic parameter report	TC[3,28] response	
	Parameter functional reporting	TC[3,37]	apply parameter functional reporting configurations	minimum	
		TC[3,38]	create a parameter functional reporting definition	by declaration	
		TC[3,39]	delete parameter functional reporting definitions	implied by TC[3,38]	

		TC[3,40]	report parameter functional reporting definitions	requires TC[3,38]	
		TM[3,41]	parameter functional reporting definition report	TC[3,40] response	
		TC[3,42]	add parameter report definitions to a parameter functional reporting definition	requires TC[3,38]	
		TC[3,43]	remove parameter report definitions from a parameter functional reporting definition	implied by TC[3,42]	
		TC[3,44]	modify the periodic generation properties of parameter report definitions of a parameter functional reporting definition	by declaration	
Service		interface	message type	required capability	implemented capability
[ST04] Parameter Statistics		TC[4,3]	reset the parameter statistics	minimum	implemented
		TC[4,1]	report the parameter statistics	minimum	implemented
		TM[4,2]	parameter statistics report	TC[4,1] response	implemented
		support for the periodic reporting of the results of the parameter statistics evaluation		by declaration	implemented
		TC[4,4]	enable the periodic parameter statistics reporting	implied by previous	implemented
		TC[4,5]	disable the periodic parameter statistics reporting	implied by previous	implemented
		TC[4,6]	add or update parameter statistics definitions	by declaration	implemented
		TC[4,7]	delete parameter statistics definitions	implied by TC[4,6]	implemented
		TC[4,8]	report the parameter statistics definitions	requires TC[4,6]	implemented
		TM[4,9]	parameter statistics definition report	TC[4,8] response	implemented
Service		interface	message type	required capability	implemented capability

[ST05] Event Report		TM[5,1]	informative event report	minimum	implemented
		TM[5,2]	low severity anomaly report	minimum	implemented
		TM[5,3]	medium severity anomaly report	minimum	implemented
		TM[5,4]	high severity anomaly report	minimum	implemented
		TC[5,5]	enable the report generation of event definitions	by declaration	implemented
		TC[5,6]	disable the report generation of event definitions	implied by TC[5,5]	implemented
		TC[5,7]	report the list of disabled event definitions	requires TC[5,5]	implemented
		TM[5,8]	disabled event definitions list report	TC[5,7] response	implemented
Service		interface	message type	required capability	implemented capability
[ST06] Memory Management	Raw data memory management	TC[6,2]	load raw memory data areas	minimum	implemented
		TC[6,5]	dump raw memory data	minimum	implemented
		TM[6,6]	dumped raw memory data report	TC[6,5] response	implemented
		TC[6,9]	check raw memory data	by declaration	implemented
		TM[6,10]	checked raw memory data report	TC[6,9] response	implemented
		TC[6,19]	load raw memory data areas by reference	by declaration	
		TC[6,20]	dump raw memory data areas to file	by declaration	
		TC[6,11]	load a raw memory atomic data area in a non-interruptible transaction	by declaration	
	Structured data memory management	TC[6,1]	load object memory data	minimum	
		TC[6,3]	dump object memory data	minimum	
		TM[6,4]	dumped object memory data report	TC[6,3] response	
		TC[6,7]	check object memory data	by declaration	

		TM[6,8]		checked object memory data report	TC[6,7] response	
		TC[6,17]	check an object memory object		by declaration	
		TM[6,18]		checked object memory object report	TC[6,17] response	
		TC[6,21]	load object memory data areas by reference		by declaration	
		TC[6,22]	dump object memory data areas to file		by declaration	
	Common memory management	TC[6,12]	abort all memory dumps		minimum	
	Memory configuration	scrubbing memories support			by declaration	
		TC[6,13]	enable the scrubbing of a memory		implied by prevously	
		TC[6,14]		disable the scrubbing of a memory	implied by prevously	
		write protecting memories support			by declaration	
		TC[6,15]	enable the write protection of a memory		implied by prevously	
		TC[6,16]		disable the write protection of a memory	implied by prevously	
Service		interface	message type		required capability	implemented capability
[ST08] Function Management		TC[8,1]	perform a function		minimum	implemented
Service		interface	message type		required capability	implemented capability
[ST09] Time Management	Time reporting	TM[9,2]			minimum	
		TM[9,3]				
		TM[9,2]	CUC time report		by declaration	
		TM[9,3]	CDS time report		by declaration	
	Time reporting control	TC[9,1]	set the time report generation rate		minimum	

Service	interface	message type	required capability	implemented capability
[ST11] Time Based Scheduling	TC[11,1]	enable the time-based schedule execution function	minimum	implemented
	TC[11,2]	disable the time-based schedule execution function	minimum	implemented
	TC[11,3]	reset the time-based schedule	minimum	implemented
	TC[11,4]	insert activities into the time-based schedule	minimum	implemented
	TC[11,20]	enable time-based sub-schedules	by declaration	
	TC[11,21]	disable time-based sub-schedules	implied by TC[11,20]	
	TC[11,18]	report the status of each time-based sub-schedule	requires TC[11,20]	
	TM[11,19]	time-based sub-schedule status report	TC[11,18] response	
	TC[11,22]	create time-based scheduling groups	by declaration	
	TC[11,23]	delete time-based scheduling groups	implied by TC[11,22]	
	TC[11,24]	enable time-based scheduling groups	implied by TC[11,22]	
	TC[11,25]	disable time-based scheduling groups	implied by TC[11,24]	
	TC[11,26]	report the status of each time-based scheduling group	requires TC[11,22]	
	TM[11,27]	time-based scheduling group status report	TC[11,26] response	
	TC[11,15]	time-shift all scheduled activities	by declaration	implemented
	TC[11,17]	summary-report all time-based scheduled activities	by declaration	implemented
	TM[11,13]	time-based schedule summary report	TC[11,17] response	implemented
	TC[11,16]	detail-report all time-based scheduled activities	by declaration	implemented
	TM[11,10]	time-based schedule detail report	TC[11,10] response	implemented

		TC[11,5]	delete time-based scheduled activities identified by request identifier	by declaration	implemented
		TC[11,7]	time-shift scheduled activities identified by request identifier	by declaration	implemented
		TC[11,12]	Summary-report time-based scheduled activities identified by request identifier	by declaration	implemented
		TM[11,13]	time-based schedule summary report	TC[11,12] response	implemented
		TC[11,9]	detail-report time-based scheduled activities identified by request identifier	by declaration	implemented
		TM[11,10]	time-based schedule detail report	TC[11,9] response	implemented
		TC[11,6]	delete the time-based scheduled activities identified by a filter	by declaration	
		TC[11,8]	time-shift the scheduled activities identified by a filter	by declaration	
		TC[11,14]	summary-report the time-based scheduled activities identified by a filter	by declaration	
		TM[11,13]	time-based schedule summary report	TC[11,14] response	
		TC[11,11]	detail-report the time-based scheduled activities identified by a filter	by declaration	
		TM[11,10]	time-based schedule detail report	TC[11,11] response	
Service		interface	message type	required capability	implemented capability
[ST12] On Board Monitoring	Parameter monitoring	TC[12,15]	enable the parameter monitoring function	minimum	
		TC[12,16]	disable the parameter monitoring function	minimum	
		TC[12,1]	enable parameter monitoring definitions	minimum	implemented
		TC[12,2]	disable parameter monitoring definitions	minimum	implemented
		TM[12,12]	check transition report	minimum	implemented
		TC[12,3]	change the maximum transition reporting delay	by declaration	implemented
		TC[12,5]	add parameter monitoring definitions	by declaration	implemented

				if TC[12,5], at least one of: TC[12,4] TC[12,6]	implied by TC[12,5]	
		TC[12,4]		delete all parameter monitoring definitions	by declaration	implemented
		TC[12,6]		delete parameter monitoring definitions	by declaration	implemented
		TC[12,7]	modify parameter monitoring definitions		by declaration	implemented
		TC[12,8]		report parameter monitoring definitions	requires TC[12,5] or TC[12,7]	implemented
		TM[12,9]		parameter monitoring definition report	TC[12,8] response	implemented
		TC[12,13]		report the status of each parameter monitoring definition	requires TC[12,1]	implemented
		TM[12,14]		parameter monitoring definition status report	TC[12,13] response	implemented
		TC[12,10]	report the out-of-limits		by declaration	implemented
		TM[12,11]		out-of-limits report	TC[12,10] response	implemented
	Functional monitoring	TC[12,17]	enable the functional monitoring function		minimum	
		TC[12,18]		disable the functional monitoring function	minimum	
		TC[12,19]	enable functional monitoring definitions		minimum	
		TC[12,20]		disable functional monitoring definitions	minimum	
		TC[12,21]	protect functional monitoring definitions		by declaration	
		TC[12,22]		unprotect functional monitoring definitions	implied by TC[12,21]	
		TC[12,23]	add functional monitoring definitions		by declaration	
		TC[12,24]		delete functional monitoring definitions	implied by TC[12,23]	

		TC[12,25]		report functional monitoring definitions	requires TC[12,23]	
		TM[12,26]		functional monitoring definition report	TC[12,25] response	
		TC[12,27]	report the status of each functional monitoring definition		by declaration	
		TM[12,28]		functional monitoring definition status report	TC[12,27] response	
Service		interface	message type		required capability	implemented capability
[ST13] Large Packet Transfer	Large packet downlink	TM[13,1]	first downlink part report (for the first part)		minimum	implemented
		TM[13,2]	intermediate downlink part report (for the intermediate parts)		minimum	implemented
		TM[13,3]	last downlink part report (for the last part)		minimum	implemented
	Large packet uplink	TC[13,9]	uplink the first part (for the first part)		minimum	implemented
		TC[13,10]	uplink an intermediate part (for the intermediate parts)		minimum	implemented
		TC[13,11]	uplink the last part (for the last part)		minimum	implemented
		TM[13,16]	large packet uplink abortion report		minimum	implemented
Service		interface	message type		required capability	implemented capability
[ST14] Real Time Forwarding Control		TC[14,1]	add report types to the application process forward-control configuration		minimum	implemented
		TC[14,2]		delete report types from the application process forward-control configuration	minimum	implemented
		TC[14,3]		report the content of the application process forward-control configuration	requires TC[14,1]	implemented
		TM[14,4]		application process forward-control configuration content report	TC[14,3] response	implemented
		capability to control, per housekeeping parameter report structure, the forwarding of housekeeping parameter reports			by declaration	

	TC[14,5]	add structure identifiers to the housekeeping parameter report forward-control configuration		implied by previous	
	TC[14,6]		delete structure identifiers from the housekeeping parameter report forward-control configuration	implied by previous	
	TC[14,7]		report the content of the housekeeping parameter report forward-control configuration	requires TC[14,5]	
	TM[14,8]		housekeeping parameter report forward-control configuration content report	TC[14,7] response	
	capability to control, per diagnostic parameter report structure, the forwarding of diagnostic parameter reports			by declaration	
	TC[14,9]	add structure identifiers to the diagnostic parameter report forward-control configuration		implied by previous	
	TC[14,10]		delete structure identifiers from the diagnostic parameter report forward-control configuration	implied by previous	
	TC[14,11]		report the content of the diagnostic parameter report forward-control configuration	requires TC[14,9]	
	TM[14,12]		diagnostic parameter report forward-control configuration content report	TC[14,11] response	
	capability to control, per event definition, the forwarding of event reports			by declaration	
	TC[14,14]	add event definition identifiers to the event report blocking forward-control configuration		implied by previous	
	TC[14,13]		delete event definition identifiers from the event report blocking forward-control configuration	implied by previous	
	TC[14,15]		report the content of the event report blocking forward-control configuration	requires TC[14,14]	
	TM[14,16]		event report blocking forward-control configuration content report	TC[14,15] response	implemented
Service	interface	message type		required capability	implemented capability

[ST15] Storage And Retrieval	Storage and retrieval	TC[15,1]	enable the storage function of packet stores		minimum	implemented
		TC[15,2]		disable the storage function of packet stores	minimum	implemented
		TC[15,14]	change the open retrieval start time tag of packet stores		minimum	implemented
		TC[15,15]	resume the open retrieval of packet stores		minimum	implemented
		TC[15,16]		suspend the open retrieval of packet stores	implied by TC[15,15]	implemented
		by-time-range retrieval function support			by declaration	
		TC[15,9]	start the by-time-range retrieval of packet stores		implied by previous	implemented
		TC[15,17]		abort the by-time-range retrieval of packet stores		implemented
		TC[15,18]	report the status of each packet store		by declaration	implemented
		TM[15,19]		packet store status report	TC[15,18] response	implemented
		TC[15,11]	delete the content of packet stores up to the specified time		by declaration	implemented
		TC[15,20]	create packet stores		by declaration	implemented
		TC[15,21]		delete packet stores	implied by TC[15,20]	implemented
		TC[15,22]		report the configuration of each packet store	requires TC[15,20]	implemented
		TM[15,23]		packet store configuration report	TC[15,22] response	implemented
		TC[15,24]		copy the packets contained in a packet store selected by time window	requires TC[15,20]	implemented
		TC[15,25]	resize packet stores		by declaration	implemented
		TC[15,26]		change a packet store type to circular	implied by TC[15.25]	implemented
		TC[15,27]		change a packet store type to bounded	implied by TC[15.25]	implemented

		TC[15,28]		change the virtual channel used by a packet store	implied by TC[15.25]	implemented
		TC[15,12]	summary-report the content of packet stores		by declaration	implemented
		TM[15,13]		packet store content summary report	TC[15,12] response	implemented
	Packet selection	TC[15,3]	add report types to the application process storage-control configuration		minimum	implemented
		TC[15,4]		delete report types from the application process storage-control configuration	minimum	implemented
		TC[15,5]		report the content of the application process storage-control configuration	requires TC[15,3]	implemented
		TM[15,6]		application process storage-control configuration content report	TC[15,5] response	implemented
		control, per housekeeping parameter report structure, the storage of housekeeping parameter reports			by declaration	
		TC[15,29]	add structure identifiers to the housekeeping parameter report storage-control configuration		implied by previous	
		TC[15,30]		delete structure identifiers from the housekeeping parameter report storage-control configuration	implied by previous	
		TC[15,35]		report the content of the housekeeping parameter report storage-control configuration	requires TC[15,29]	
		TM[15,36]		housekeeping parameter report storage-control configuration content report	TC[15,36] response	
		control, per diagnostic parameter report structure, the storage of diagnostic parameter reports			by declaration	
		TC[15,31]	add structure identifiers to the diagnostic parameter report storage-control configuration		implied by previous	
		TC[15,32]		delete structure identifiers from the diagnostic parameter report storage-control configuration	implied by previous	

		TC[15,37]		report the content of the diagnostic parameter report storage-control configuration		requires TC[15,31]	
		TM[15,38]			diagnostic parameter report storage-control configuration content report	TC[15,37] response	
		control, per event definition, the storage of event reports				by declaration	
		TC[15,34]	add event definition identifiers to the event report blocking storage-control configuration			implied by previous	
		TC[15,33]		delete event definition identifiers from the event report blocking storage-control configuration			
		TC[15,39]		report the content of the event report blocking storage-control configuration		requires TC[15,33]	
		TM[15,40]			event report blocking storage-control configuration content report	TC[15,39] response	
Service		interface	message type			required capability	implemented capability
[ST17] Test		TC[17,1]	perform an are-you-alive connection test			minimum	implemented
		TM[17,2]		are-you-alive connection test report		TC[17,1] response	implemented
		TC[17,3]	perform an on-board connection test			minimum	implemented
		TM[17,4]		on-board connection test report		TC[17,4] response	implemented
Service		interface	message type			required capability	implemented capability

[ST18] On-board Operations Procedure	OBCP management		at least one of: TC[18,1] TC[18,13] TC[18,19]	minimum	
		TC[18,1]	direct-load an OBCP	by declaration	
		TC[18,13]	load an OBCP by reference	by declaration	
		TC[18,2]	unload an OBCP	implied by TC[18,1] or TC[18,13]	
		TC[18,3]	activate an OBCP	minimum	
		TC[18,19]	load by reference and activate an OBCP	by declaration	
		TC[18,4]	stop an OBCP	minimum	
		TC[18,20]	stop and unload an OBCP	by declaration	
		TC[18,12]	abort an OBCP	minimum	
		TC[18,17]	abort all OBCPs and report	by declaration	
		TM[18,18]	aborted OBCP report	TC[18,17] response	
		TC[18,8]	report the execution status of each OBCP	minimum	
		TM[18,9]	OBCP execution status report	TC[18,8] response	
		TC[18,5]	suspend an OBCP	by declaration	
		TC[18,6]	resume an OBCP	implied by TC[18,5]	
		TC[18,14]	activate and execute one OBCP step	by declaration	
		TC[18,15]	resume and execute one OBCP step	implied by TC[18,14]	
		TC[18,7]	communicate parameters to an OBCP	by declaration	
		TC[18,16]	set the observability level of OBCPs	by declaration	

	OBCP engine management	TC[18,21]	start the OBCP engine	minimum	
		TC[18,22]	stop the OBCP engine	minimum	
Service		interface	message type	required capability	implemented capability
[ST19] Event Action		TC[19,8]	enable the event-action function	minimum	implemented
		TC[19,9]	disable the event-action function	minimum	implemented
		TC[19,4]	enable event-action definitions	minimum	implemented
		TC[19,5]	disable event-action definitions	minimum	implemented
		TC[19,1]	add event-action definitions	minimum	implemented
			at least one of: TC[19,2] TC[19,3]	implied by TC[19,1]	
		TC[19,2]	delete event-action definitions	by declaration	implemented
		TC[19,3]	delete all event-action definitions	by declaration	implemented
		TC[19,6]	report the status of each event-action definition	by declaration	implemented
		TM[19,7]	event-action status report	TC[19,6] response	implemented
		TC[19,10]	report event-action definitions	requires TC[19,1]	implemented
		TM[19,11]	event-action definition report	TC[19,10] response	implemented
Service		interface	message type	required capability	implemented capability
[ST20] Parameter Management		TC[20,1]	report parameter values	minimum	implemented
		TM[20,2]	parameter value report	TC[20,1] response	implemented
		TC[20,3]	set parameter values	by declaration	implemented

	TC[20,4]	change raw memory parameter definitions		by declaration	
	TC[20,5]	change object memory parameter definitions		by declaration	
	TC[20,6]		report parameter definitions	requires TC[20,4] or TC[20,5]	
	TM[20,7]		parameter definition report	TC[20,6] response	
Service	interface	message type		required capability	implemented capability
[ST21] Request Sequencing	at least one of: TC[21,1] TC[21,2] TC[21,8]			minimum	
	TC[21,1]	direct-load a request sequence		by declaration	implemented
	TC[21,2]	load a request sequence by reference		by declaration	implemented
	TC[21,3]		unload a request sequence	implied by TC[21,1] or TC[21,2]	implemented
	TC[21,8]	load by reference and activate a request sequence		by declaration	implemented
	TC[21,4]	activate a request sequence		minimum	implemented
	TC[21,5]	abort a request sequence		minimum	implemented
	TC[21,13]	abort all request sequences and report		by declaration	implemented
	TM[21,14]		aborted request sequence report	TC[21,13] response	implemented
	TC[21,6]	report the execution status of each request sequence		by declaration	implemented
	TM[21,7]		request sequence execution status report	TC[21,6] response	implemented
	TC[21,9]	checksum a request sequence		by declaration	implemented
	TM[21,10]		request sequence	TC[21,9] response	implemented

			checksum report			
	TC[21,11]	report the content of a request sequence			by declaration	implemented
	TM[21,12]		request sequence content report		TC[21,11] response	implemented
Service	interface	message type			required capability	implemented capability
[ST22] Position-based Scheduling	TC[22,1]	enable the position-based schedule execution function			minimum	
	TC[22,2]		disable the position-based schedule execution function		minimum	
	TC[22,28]	set the orbit number			by declaration	
	TC[22,3]	reset the position-based schedule			minimum	
	TC[22,4]	insert activities into the position-based schedule			minimum	
	TC[22,20]	enable position-based sub-schedules			by declaration	
	TC[22,21]		disable position-based sub-schedules		implied by TC[22,20]	
	TC[22,18]	report the status of each position-based sub-schedule			by declaration	
	TM[22,19]		position-based sub-schedule status report		TC[22,18] response	
	TC[22,22]	create position-based scheduling groups			by declaration	
	TC[22,23]		delete position-based scheduling groups		implied by TC[22,22]	
	TC[22,24]		enable position-based scheduling groups		implied by TC[22,22]	
	TC[22,25]			disable position-based scheduling groups	implied by TC[22,24]	
	TC[22,26]		report the status of each position-based scheduling group		requires TC[22,22]	
	TM[22,27]			position-based scheduling group status report	TC[22,26] response	

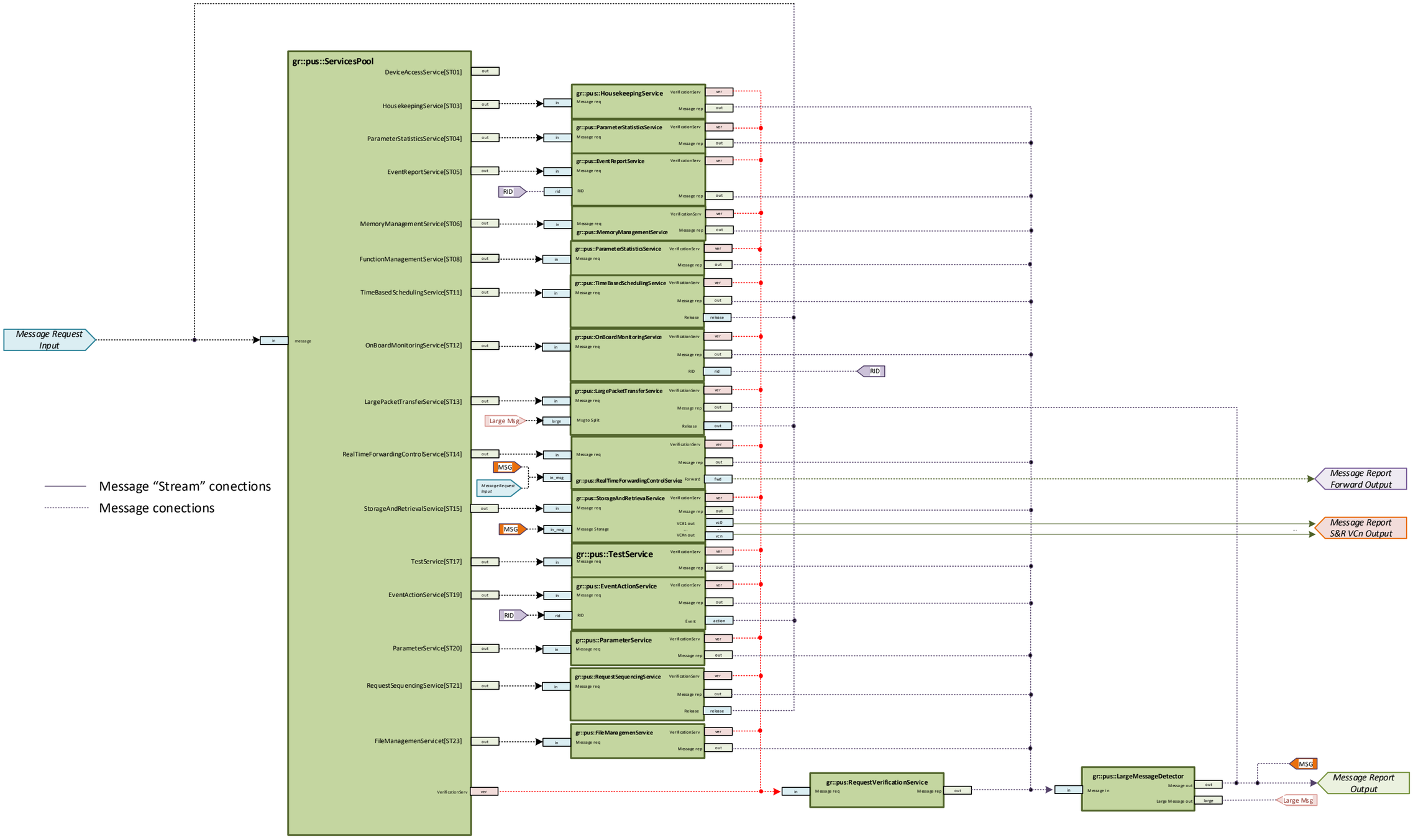
	TC[22,15]	position-shift all scheduled activities				by declaration	
	TC[22,17]	summary-report all position-based scheduled activities				by declaration	
	TM[22,13]		position-based schedule summary report			TC[22,17] response	
	TC[22,16]	detail-report all position-based scheduled activities				by declaration	
	TM[22,10]		position-based schedule detail report			TC[22,16] response	
	TC[22,5]	delete position-based scheduled activities identified by request identifier				by declaration	
	TC[22,7]	position-shift scheduled activities identified by request identifier				by declaration	
	TC[22,12]	summary-report position-based scheduled activities identified by request identifier				by declaration	
	TM[22,13]		position-based schedule summary report				
	TC[22,9]	detail-report position-based scheduled activities identified by request identifier				by declaration	
	TM[22,10]		position-based schedule detail report				
	TC[22,6]	delete the position-based scheduled activities identified by a filter				by declaration	
	TC[22,8]	position-shift the scheduled activities identified by a filter				by declaration	
	TC[22,14]	summary-report the position-based scheduled activities identified by a filter				by declaration	
	TM[22,13]		position-based schedule summary report			TC[22,14] response	
	TC[22,11]	detail-report the position-based scheduled activities identified by a filter				by declaration	
	TM[22,10]		position-based schedule detail report			TC[22,11] response	

Service		interface	message type	required capability	implemented capability
[ST23] File Management	File handling	TC[23,1]	create a file	minimum	implemented
		TC[23,2]	delete a file	minimum	implemented
		TC[23,3]	report the attributes of a file	minimum	implemented
		TM[23,4]	file attribute report	TC[23,3] response	implemented
		TC[23,5]	lock a file	by declaration	implemented
		TC[23,6]	unlock a file	implied by TC[23,5]	implemented
		TC[23,7]	find files	by declaration	implemented
		TM[23,8]	found files report	TC[23,7] response	implemented
		TC[23,9]	create a directory	by declaration	implemented
		TC[23,10]	delete a directory	implied by TC[23,9]	implemented
		TC[23,11]	rename a directory	implied by TC[23,9]	implemented
		TC[23,12]	summary-report the content of a repository	by declaration	implemented
		TM[23,13]	repository content summary report	TC[23,12] response	implemented
	File copy	TC[23,14]	copy a file	minimum	implemented
		TC[23,15]	move a file	by declaration	implemented
		TC[23,16]	suspend file copy operations	by declaration	
		TC[23,17]	resume file copy operations	implied by TC[23,16]	
		TC[23,19]	suspend all file copy operations involving a repository path	by declaration	

		TC[23,20]		resume all file copy operations involving a repository path	implied by TC[23,19]	
		TC[23,18]	abort file copy operations		by declaration	
		TC[23,21]	abort all file copy operations involving a repository path		by declaration	
		TC[23,22]	enable the periodic reporting of the file copy status		by declaration	
		TC[23,24]		disable the periodic reporting of the file copy status	implied by TC[23,22]	
		TM[23,23]		file copy status report	TC[23,22] response	

The gr-pus implementation is based on several OOT blocks inter-connected to provides the PUS functionalities as follow:

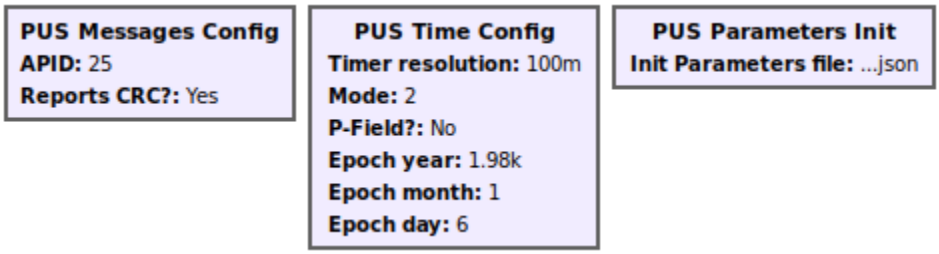
- Any message request will enter into the Service Pool block for integrity check and target service distribution, this includes, also, any message request generated by event action, time based scheduling and request sequencing services
- Service pool will forward the message requests to the target service, whitch will execute any valid request and output, if any, the message reports
- All reports pass through a large message detector block, which will extract the larges ones and forward these to the large packet transfer service to be splitted
- The reports will be sent to the Store and Retrieval and, optionally, to Real Time Forwarding Services, those services will check out if store or forward are required, otherwise will ignore these reports
- The Real Time Forwarding Services could be connected to the message requests input to forward any message with another AppID than the one for used by the implementation (ie another subsystem). This Service does a basic verification on the messages to forward: check for the primary and secondary headers just to know if it is a PUS packet and has a valid the AppID, Service Type and Message Type, but doesn't check size and CRC, it is up to the destination AppID to do that
- Tracking of the message requests execution status will be done by each service and service pool, whitch will send to the request verification service the status of any request according to the ack flags setting and errors. The verification servic willl generated the reports
- The timer tick will periodically call the services callbacks such as the Housekeeping, Parameter Statistics, Time Based Scheduling, On Board Monitoring, Storage And Retrieval and Request Sequencing services in order to update any action required by those services
- The On Board Monitoring, upon a parameter out of expected value detection, will generated a message with the RID information for the Event Report and Event Action services, those services will generated the report and trigger, if any, the action
- If any report is forward through the Real Time Forwarding Service, it will send through this service forward output
- When retrieve stored data from the Store and Retrieval Service, those stored reports will be ouputed thru the corresponding Virtual channel, but at certain bit rate, otherwise all messages will be outputed as fast as the software could handle them. The Store and Retrieval Service will send an integer number of messages (N messages size ≈ bitrate * timer tick resolution) and it will wait certaint time until send next messages batch to keep the overal output bit rate.


Message Request Input
gr::pus::ServicesPool
DeviceAccessService[ST01]
HousekeepingService[ST03]
ParameterStatisticsService[ST04]
EventReportService[ST05]
MemoryManagementService[ST06]
FunctionManagementService[ST08]
TimeBasedSchedulingService[ST11]
OnBoardMonitoringService[ST12]
LargePacketTransferService[ST13]
RealTimeForwardingControlService[ST14]
StorageAndRetrievalService[ST15]
TestService[ST17]
EventActionService[ST19]
ParameterService[ST20]
RequestSequencingService[ST21]
FileManagemenServicet[ST23]
VerificationServ
message
out
in
rid
large
in_msg
ver
release
action
fwd
vc0
vcn
RID
MSG
Large Msg
MessageRequest Input
gr::pus::HousekeepingService
gr::pus::ParameterStatisticsService
gr::pus::EventReportService
gr::pus::MemoryManagementService
gr::pus::TimeBasedSchedulingService
gr::pus::OnBoardMonitoringService
gr::pus::LargePacketTransferService
gr::pus::RealTimeForwardingControlService
gr::pus::StorageAndRetrievalService
gr::pus::TestService
gr::pus::EventActionService
gr::pus::ParameterService
gr::pus::RequestSequencingService
gr::pus::FileManagemenService
Message req
Message rep
Release
Msg to Split
Forward
Message Storage
VC#1 out
VCn out
Event
gr::pus::RequestVerificationService
gr::pus::LargeMessageDetector
Message in
Message out
Large Message out
Message Report Forward Output
Message Report S&R VCn Output
Message Report Output
Message “Stream” conections
Message conections

2.1 INIT

In order to initialize the gr-pus, a group of helpers OOT blocks are available

- PUS message config will initializate the APID used by the application running the gr-pus and if the message reports will use the CRC field
- PUS Time Config will initializate the timer tick resolución (in seconds), and the time format CUC1, CUC2 or CDS and if pfield is present (only CUC1 and 2 without pfield work for this release). For CUC2, this block will allows to configure the epoch
- PUS Parameters Init will initializate the parameters pool, see next chapter



2.2 PARAMETERS

Paramaters are handled by a singleton class called ParameterPool. This class has a parameters map matching ID with parameter type and its value.

Parameter initialization are performed using the the initializeParameterMap method of the ParameterPool class, in the example and tests this method is called usign the ParametersInit block with a json file with all parameters data.

The json file has next format (see examples folder for all json files):

```
{
   "parameters": [
      {
         "id": 1,
         "type": "uint8",
         "default": 8
      },
      {
         "id": 2,
         "type": "sint16",
         "default": -23243
      },
.....
   ]
```

}

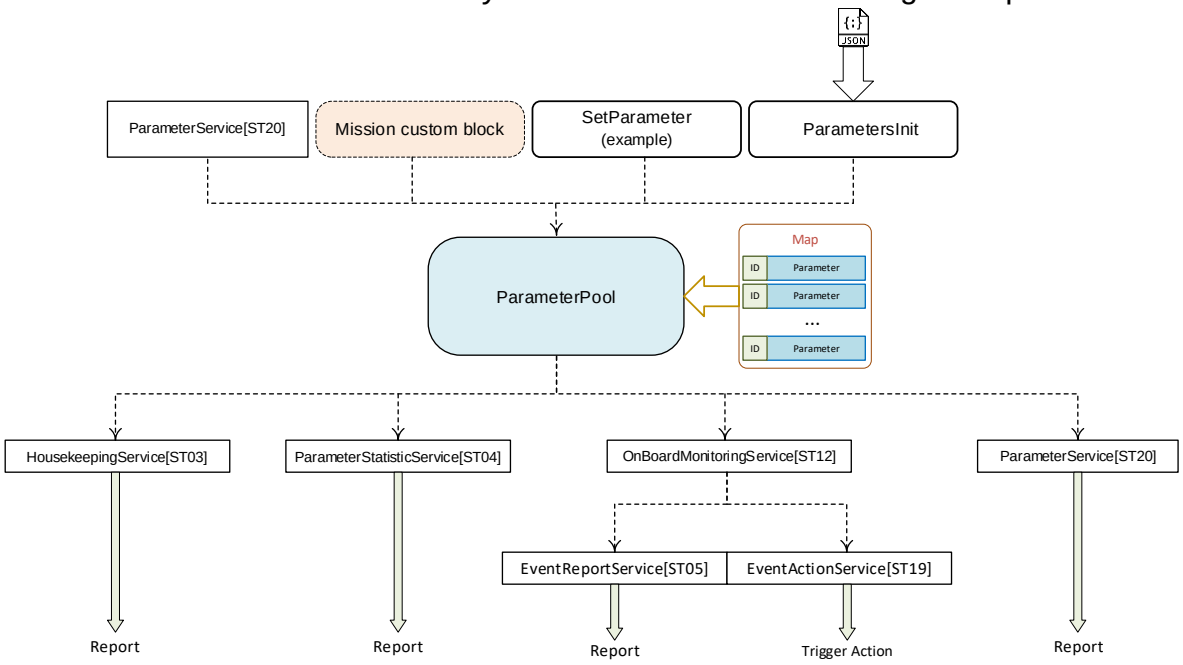
No parameter could be added by any method other than the initializeParameterPool, and this method will clean up any parameters before load the dataset, then the full parameters dataset shall be included into the json file, whitch means define each parameter ID, its type and its default (startup) value

The parameterers value could be set or get remotely usign the Parameter Service ST[20] subservices, or “locally” using the SetParameter block.

In addition, each parameter includes an optional callback fuction and each time the parameter value is set, it will call this function

The SetParameter block is an example helper OOT block intented to show how to set a parameter value during runtime (and used in testing) and it will print a message each time the parameter is updated, but a specific OOT block/code will be needed for each implementation matching any configuration or status variable from hardware/software with the ParameterPool parameters

The ParameterPool will be used by the rest of PUS Services to get/set paramaters



2.3 TIME AND TIMMING

The time is handled by a singleton class called TimeProvider. This class includes a timer tick responsable to call the Services requiring time Schedule activities, such as

the Housekeeping Service. The timer tick is in sync with each seconds start, meaning the first tick will be executed when a second start and then it will keep tracking of passing secods sync

Currently only CUC1/CUC2 time formats with 4 octects (uint32_t) of time and no fractional time is supported

Time initialization are performed using the the TimeConfig block allowing to set the Epoch for CUC2 and timer tick resolution in secods

2.4 ERRORS

Next errors codes are used in the Request Verification Service reports, not all of them are implemented, but they are kept for compatibility with A*cubeSAT*[1]

[1] This project uses as baseline the AcubeSAT PUS service implementation from Octuber 2023 and added additional services and complementary subtype, adding later on the FileManagement Service base don the AcubeSat one (not included in the October 2023 release), then, no all the errors codes will match with the AcubeSAT latest releases

Error Type	Error Code	Error Description
InternalErrorType	UnknownInternalError = 0	
	MessageTooLarge = 1	While writing (creating) a message, an amount of bytes was tried to be added but resulted in failure, since the message storage was not enough.
	TooManyBitsAppend = 2	Asked to append a number of bits larger than supported
	ByteBetweenBits = 3	Asked to append a byte, while the previous byte was not complete
	StringTooLarge = 4	A string is larger than the largest allowed string
	UnacceptablePacket = 5	An error in the header of a packet makes it unable to be parsed
	InvalidDate = 6	A date that isn't valid according to the Gregorian calendar or cannot be parsed by the TimeHelper
	UnknownMessageType = 7	Asked a Message type that doesn't exist
	InvalidSpareBits = 8	Asked to append unnecessary spare bits
	OtherMessageType = 9	A function received a Message that was not of the correct type
	MapFull = 10	Attempt to insert new element in a full map ST[08]
	NestedMessageTooLarge = 11	A Message that is included within another message is too large
	InvalidTimeWindowType = 12	Request to copy packets in a time window, whose type is not recognized (ST(15)).
	NonExistentHousekeeping = 13	A request to access a non existing housekeeping structure in ST[03]
	NonExistentParameter = 14	Attempt to access an invalid parameter in ST[03]
	InvalidTimeStampInput = 15	Invalid TimeStamp parameters at creation
	ElementNotInArray = 16	A requested element is not found
	TimeStampOutOfBounds = 17	Timestamp out of bounds to be stored or converted
	EventReportTypeUnknown = 48	In the Event Report Service, an unknown type report request was made
AcceptanceErrorType The error code for failed acceptance reports, as specified in ECSS 6.1.4.3d	UnknownAcceptanceError = 0	
	StringTooShort = 4	Cannot read a string, because it is larger than the largest allowed string
	UnacceptableMessage = 5	Cannot parse a Message, because there is an error in its secondary header
	IllegalAPID = 0	The received message size does not contain a valid APID

	InvalidLength = 1	The received message size does not contain a valid data lenght
	InvalidChecksum = 2	The received message CRC/checksum is invalid
ExecutionStartErrorType The error code for failed start of execution reports, as specified in ECSS 5.3.5.2.3g	UnknownExecutionStartError = 0	
	EventDefinitionIDExistsError = 1	In the Event Action Service, in the addEventActionDefinition function an attempt was made to add an event Action Definition with an eventDefinitionID that exists
	EventActionEnabledError = 2	In the Event Action Service, in the addEventActionDefinition function an attempt was made to add an event Action Definition that is already enabled
	EventActionDeleteEnabledDefinitionError = 3	In the Event Action Service, in the deleteEventActionDefinition function, an attempt was made to delete an event action definition that was enabled
	EventActionUnknownEventActionDefinitionError = 4	In the Event Action Service, an access attempt was made to an unknown event action definition
	EventActionUnknownEventActionDefinitionIDError = 5	EventAction refers to the service, EventActionIDefinitionID refers to the variable In the Event Action Service, an access attempt was made to an unknown eventDefinitionID
	SubServiceExecutionStartError = 6	Unable to start execution of subservice
	InstructionExecutionStartError = 7	Unable to start execution of instruction
	SetNonExistingParameter = 8	Attempt to change the value of a non existing parameter (ST[20])
	GetNonExistingParameter = 9	Attempt to access a non existing parameter (ST[20])
	NonExistingPacketStore = 10	Attempt to access a packet store that does not exist (ST[15])
	SetPacketStoreWithOpenRetrievalInProgress = 11	Attempt to change the start time tag of a packet store, whose open retrieval status is in progress (ST[15])
	SetPacketStoreWithByTimeRangeRetrieval = 12	Attempt to resume open retrieval of a packet store, whose by-time-range retrieval is enabled (ST[15])
	GetPacketStoreWithByTimeRangeRetrieval = 13	Attempt to access a packet with by-time range retrieval enabled (ST[15])
	GetPacketStoreWithOpenRetrievalInProgress = 14	Attempt to start the by-time-range retrieval of packet store, whose open retrieval is in progress (ST[15])
	ByTimeRangeRetrievalAlreadyEnabled = 15	Attempt to start by-time-range retrieval when its already enabled (ST[15])
	AlreadyExistingPacketStore = 16	Attempt to create packet store, whose ID already exists (ST[15])
	MaxNumberOfPacketStoresReached = 17	Attempt to create packet store, when the max number of packet stores is already reached (ST[15])

	GetPacketStoreWithStorageStatusEnabled = 18	Attempt to access a packet store with the storage status enabled (ST[15])
	DeletionOfPacketWithByTimeRangeRetrieval = 19	Attempt to delete a packet whose by time range retrieval status is enabled (ST[15])
	DeletionOfPacketWithOpenRetrievalInProgress = 20	Attempt to delete a packet whose open retrieval status is in progress (ST[15])
	InvalidTimeWindow = 21	Requested a time window where the start time is larger than the end time (ST[15])
	DestinationPacketStoreNotEmtpy = 22	Attempt to copy a packet store to a destination packet store that is not empty (ST[15])
	InvalidReportingRateError = 23	Attempt to set a reporting rate which is smaller than the parameter sampling rate (ST[04]).
	EventActionDefinitionsMapIsFull = 24	Attempt to add definition to the struct map but its already full.(ST[19])
	RequestedNonExistingStructure = 25	Attempt to report/delete non existing housekeeping structure (ST[03])
	RequestedAlreadyExistingStructure = 26	Attempt to create already created structure (ST[03])
	RequestedDeletionOfEnabledHousekeeping = 27	Attempt to delete structure which has the periodic reporting status enabled (ST[03]) as per 6.3.3.5.2(d-2)
	AlreadyExistingParameter = 28	Attempt to append a new parameter ID to a housekeeping structure, but the ID is already in the structure (ST[03])
	RequestedAppendToEnabledHousekeeping = 29	Attempt to append a new parameter id to a housekeeping structure, but the periodic generation status is enabled (ST[03])
	ExceededMaxNumberOfHousekeepingStructures = 30	Attempt to create a new housekeeping structure in Housekeeping Service, when the maximum number of housekeeping structures is already reached (ST[03])
	ExceededMaxNumberOfSimplyCommutatedParameters = 31	Attempt to add a new simply commutated parameter in a specific housekeeping structure, but the maximum number of simply commutated parameters for this structure is already reached (ST[03])
	InvalidSamplingRateError = 32	Attempt to set a reporting rate which is smaller than the parameter sampling rate (ST[04])
	MaxStatisticDefinitionsReached = 33	Attempt to add new statistic definition but the maximum number is already reached (ST[04])
	InvalidVirtualChannel = 34	Attempt to set the virtual channel of a packet store to a invalid value (ST[15])
	DeletionOfPacketStoreWithStorageStatusEnabled = 35	Attempt to delete a packet store, whose storage status is enabled (ST[15])

CopyOfPacketsFailed = 36	Attempt to copy packets from a packet store to another, but either no packet timestamp falls inside the specified timestamp, or more than one boolean argument were given as true in the 'copyPacketsTo' function (ST[15])
UnableToHandlePacketStoreSize = 37	Attempt to set a packet store size to a value that the available memory cannot handle (ST[15]).
InvalidRequestToDeleteAllParameterMonitoringDefinitions = 38	Attempt to delete all parameter monitoring definitions but the Parameter Monitoring Function Status is enabled.
InvalidRequestToDeleteParameterMonitoringDefinition = 39	Attempt to delete one parameter monitoring definition but its Parameter Monitoring Status is enabled.
AddAlreadyExistingParameter = 40	Attempt to add a parameter that already exists to the Parameter Monitoring List.
ParameterMonitoringListIsFull = 41	Attempt to add a parameter in the Parameter Monitoring List but it's full
HighLimitIsLowerThanLowLimit = 42	Attempt to add or modify a limit check parameter monitoring definition, but the high limit is lower than the low limit.
HighThresholdIsLowerThanLowThreshold = 43	Attempt to add or modify a delta check parameter monitoring definition, but the high threshold is lower than the low threshold.
ModifyParameterNotInTheParameterMonitoringList = 44	Attempt to modify a non existent Parameter Monitoring definition.
DifferentParameterMonitoringDefinitionAndMonitoredParameter = 45	Attempt to modify a parameter monitoring definition, but the instruction refers to a monitored parameter that is not the one used in that parameter monitoring definition.
GetNonExistingParameterMonitoringDefinition = 46	Attempt to get a parameter monitoring definition that does not exist.
ReportParameterNotInTheParameterMonitoringList = 47	Request to report a non existent parameter monitoring definition.
AllServiceTypesAlreadyAllowed = 48	Attempt to add a new report type, when the addition of all report types is already enabled in the Application Process configuration (ST[14])
MaxReportTypesReached = 49	Attempt to add a new report type, when the max number of reports types allowed per service type definition in the Application Process configuration is already reached (ST[14])
MaxServiceTypesReached = 50	Attempt to add a new service type, when the max number of service types allowed per application process definition in the Application Process configuration is already reached (ST[14])
NotControlledApplication = 51	Attempt to add a report/event definition/housekeeping report type, when the specified application process ID is not controlled by the Service (ST[14])

ParameterValueMissing = 52	Parameter is requested, but the provider of the parameter value does not exist yet
ParameterReadOnly = 53	Attempted to write to a read-only parameter
ParameterWriteOnly = 54	Attempted to read from a write-only parameter
NonExistentReportTypeDefinition = 55	Attempt to access a non-existing report type definition, from the application process configuration (ST[14])
NonExistentServiceTypeDefinition = 56	Attempt to access a non-existing service type definition, from the application process configuration (ST[14])
NonExistentApplicationProcess = 57	Attempt to access a non-existing application process definition, from the application process configuration (ST[14])
MissingMessageData = 58	Missing data in the message request
LargePacketTransactionIDAlreadyExist = 59	Received an Large Packet Transfer first part, but already existe in the uplink buffer (ST[13])
LargePacketTransactionIDNonExist = 60	Received an Large Packet Transfer intermediate or end part, but non exists in the uplink buffer (ST[13])
LargePacketTransactionWrongSequenceNumber = 61	Wrong number in Large Packet Transfer first part(ST[13])
NonExistingFunction = 62	Attempt to call a non existing Function (ST[8])
ParameterUnknown = 63	Parameter is requested, but the provider of the parameter value does not exist yet
InvalidEventSelection = 64	In the Event Report Service, in the enable/disable report generation, an attempt was made to enable/disable an non existing event
MaxReportTypesReachedInProgress = 65	Attempt to add a new report type, when the max number of reports types allowed per service type definition in the Application Process configuration is already reached (ST[14])
setNonExistingParameterMonitoringCheckDefinition = 66	Attempt to add an unknown parameter monitoring check definition (ST[12]).
setInvalidMonitoringCheckDefinition = 67	Attempt to add a parameter monitoring check definition with low check values higher than high check values(ST[12]).
setInvalidMonitoringCheckDefinitionAlreadyExists = 68	Attempt to add a parameter monitoring check definition that already exists (ST[12]).
deleteMonitoringDefinitionEnabled = 69	Attempt to delete a parameter monitoring definition that is enabled (ST[12]).
invalidMonitoringCheckDefinitionWrongParameter = 70	Attempt to modify a non existing parameter monitoring check definition (ST[12]).

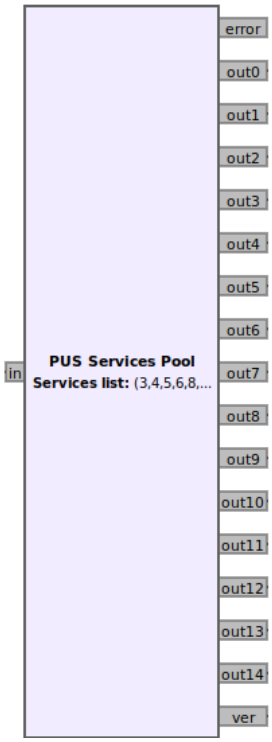
	alreadyExistingReportTypeDefinition = 71	Attempt to add an already existing report type definition, in the application process configuration (ST[14])
	undefinedMemory = 72	Attempt to access to an undefined memory type in the memory management (ST[06])
	readOnlyMemory = 73	Attempt to write in a read only memory (ST[06])
	AlreadyExistingSequenceStore = 74	Attempt to write in an existing sequence store (ST[21])
	MaxNumberOfSequenceStoresReached = 75	Attempt to create a sequence, when the max number of seq stores is already reached (ST[21])
	NumberOfSequenceActivitiesOverflow = 76	Attempt to create a sequence with a number of activiies higher than the allowed one (ST[21])
	NonExistingSequenceStore = 77	Attempt to access a non existing sequence store (ST[21])
	UnderExecutionSequenceStore = 78	Attempt to unload an under execution sequence store (ST[21])
	FileNotFound = 79	Attempt to access to a non existing file
	MaxActivatedSequencesReach = 80	Maximum number of activated sequences reached by sequence store (ST[21])
	AlreadyInactiveSequence = 81	Attempt to abort one already inactive sequence in sequence store (ST[21])
	SizeOfFileIsOutOfBounds = 82	Attempt to create a file large than allowed one (ST[23])
	ObjectPathIsInvalid = 83	Attempt to access to an object, whitch path is invalid (ST[23])
ExecutionProgressErrorType The error code for failed progress of execution reports, as specified in ECSS 5.3.5.2.3g	UnknownExecutionProgressError = 0	
	AlreadyExistingStorageDefinition = 48	Attempt to Add a new storage definition, but the definition is already in the structure (ST[15])
	GetNonExistingStorageDefinition = 49	Attempt to Remove a non existing storage definition (ST[15])
ExecutionCompletionErrorType The error code for failed completion of execution reports, as specified in ECSS 5.3.5.2.3g	UnknownExecutionCompletionError = 0	
	ChecksumFailed = 1	Checksum comparison failed
	AddressOutOfRange = 2	Address of a memory is out of the defined range for the type of memory
	FileAlreadyExists = 3	File already exists, thus can't be created again
	ObjectDoesNotExist = 4	The requested object does not exist
	AttemptedDeleteOnLockedFile = 5	A delete file command was requested on a file that is locked

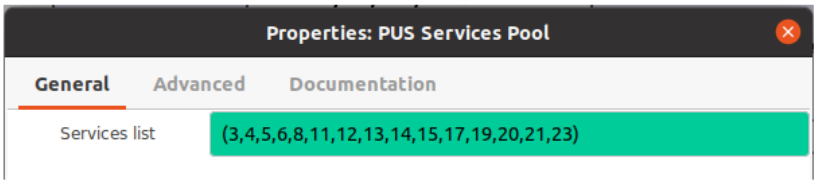
	AttemptedDeleteOnDirectory = 6	A delete file command was requested on a directory
	UnknownFileDeleteError = 7	The filesystem reported an error during file deletion
	AttemptedReportAttributesOnDirectory = 8	A report file attributes command was requested on a directory
	DirAlreadyExists = 9	Dir already exists, thus can't be created again
	UnknownDirDeleteError = 10	The filesystem reported an error during directory deletion
	UnknownDirRenameError = 11	The filesystem reported an error during directory deletion
	UnknownFileCopyError = 12	The filesystem reported an error during file copy
	UnknownFileMoveError = 13	The filesystem reported an error during file move
	UnknownFileAttributeError = 14	The filesystem reported an error during file attribute retrieval
	AttemptedMoveALockedFile = 15	A move a file command was requested on a file that is locked
	AttemptedDeleteANonEmptyDir = 16	A move a file command was requested on a file that is locked
RoutingErrorType The error code for failed routing reports, as specified in ECSS 6.1.3.3d	UnknownRoutingError = 0	

2.5 SERVICE POOL

The ***PUS Services Pool*** block will check any input message request for errors, including size and CRC and, if it is valid message request and the destination service exists, it will forward the message to the target output service.

This block will no check the message subtype integrity and request specific data size other than the size field in the primary header match with the message real size, the checkings related with service subtype will be performed by each service





Parameters

(*R*): *Run-time adjustable*

Services list

Set the vector with the services list for message forwarding, it will be matched against output out0, out1, etc

Messages

In

The message input

Out0…n

The message output for forward services, matched against the Service list

ver

The message verification output, if an error is detected, an output message for the Request Verification Service ST[01] will be addressed

2.6 ST[01] REQUEST VERIFICATION

The ***ST[01] Verification Service*** block will received from all services the request verification messages and generate the ST[01] Request Verification Services reports at its output

Parameters

(*R*): *Run-time adjustable*

Messages

In

The verification message input. It is composed by the original message causing the verification request plus metadata appended into the message:

pmt::intern("req") , pmt::from_long(requestVerificationServiceReportSubType)

if it is an error report:

pmt::intern("error_type"), pmt::from_long(error_code)

for subtype 5 and 6:

pmt::intern("step_id"), pmt::from_long(step_number)

Out

The message report output

Subtypes requests

TM[1,1] successful acceptance verification report

Successful acceptance verification report

request ID					
packet version number	packet ID			packet sequence control	
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)

TM[1,2] failed acceptance verification report

Failed acceptance verification report

request ID						failure notice	
packet version number	packet ID			packet sequence control		code	data
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count		
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)	enumerated	deduced

deduced presence

uint16 Non present

TM[1,3] successful start of execution verification report

Successful start of execution verification report

request ID					
packet version number	packet ID			packet sequence control	
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)

TM[1,4] failed start of execution verification report

Failed start of execution verification report

request ID						failure notice	
packet version number	packet ID			packet sequence control		code	data
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count		
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)	enumerated	deduced
							deduced presence
						uint16	Non present

TM[1,5] successful progress of execution verification report

Successful progress of execution verification report

request ID						step ID
packet version number	packet ID			packet sequence control		
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count	
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)	enumerated
						uint8

TM[1,6] failed progress of execution verification report

Failed progress of execution verification report

request ID						step ID	failure notice	
packet version number	packet ID			packet sequence control			code	data
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count			
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)	enumerated	enumerated	deduced
								deduced presence
						uint8	uint16	Non present

TM[1,7] successful completion of execution report

Successful completion of execution verification report

request ID					
packet version number	packet ID			packet sequence control	
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)

TM[1,8] failed completion of execution verification report

Failed completion of execution verification report

request ID						failure notice	
packet version number	packet ID			packet sequence control		code	data
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count		
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)	enumerated	deduced

deduced presence

uint16 Non present

TM[1,10] failed routing verification report

Failed routing verification report

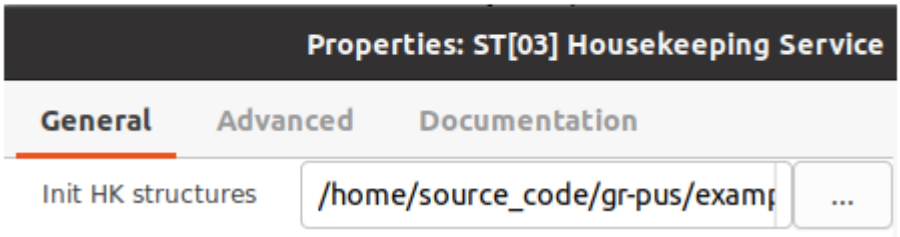
request ID						failure notice	
packet version number	packet ID			packet sequence control		code	data
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count		
enumerated (3 bits)	enumerated (1 bit)	Boolean (1 bit)	enumerated (11 bits)	enumerated (2 bits)	unsigned integer (14 bits)	enumerated	deduced

deduced presence

uint16 Non present

2.7 ST[03] HOUSEKEEPING

The ***ST[03] Housekeeping Service*** block will received all message request at its input port and if those request are for service ST[03] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Init HK structures

Path to the json file with the start up housekeeping structures, left empty if no init required

Messages

In

The message requests input

Out

The message report output

ver

The message verification output, if an error is detected, an output message for the Request Verification Service ST[01] will be addressed

The json init file has next format:

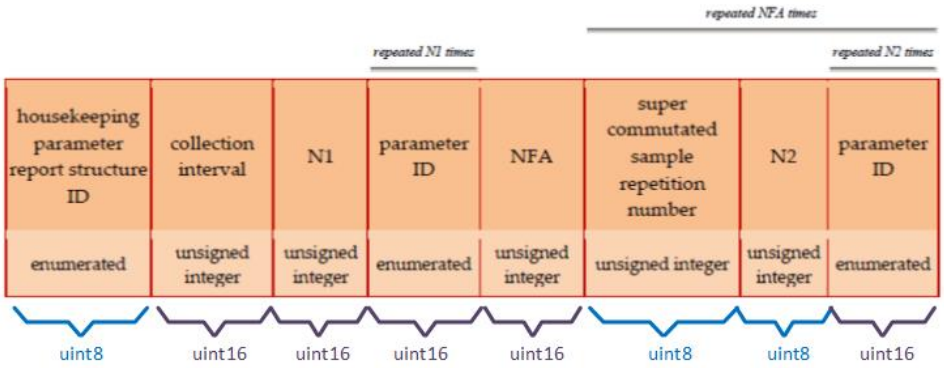
```
{
    "housekeeping": [
        {
            "id": 1,
            "interval": 50,
            "periodicEnabled": false,
            "numParams": 2,
            "Params": [
                3,
                16
            ]
        },
        {
            "id": 4,
            "interval": 50,
            "periodicEnabled": false,
            "numParams": 6,
            "Params": [
                5,
                1,
                16,
                34,
                11,
                31
            ],
            "SuperArrays": [
                {
                    "superRepetition": 4,
                    "numParams": 3,
                    "Params": [
                        17,
                        9,
                        3
                    ]
                },
                {
                    "superRepetition": 5,
                    "numParams": 1,
                    "Params": [
                        15
                    ]
                }
            ]
        },
```

The interval field is expresed in timer tick values, then if the timer generates a tick each 100ms, then “interval” = 50 represents 5 sec.
The super conmutated parameters repetition is expressed in times related to the interval duration, then for “interval” = 50 with a timer tick each 100ms, whitch represents 5 sec, a “superRepetition” = 5 means the super conmutated parameters will be read 5 times in these 5 sec, then once per second.
The interval/superRepetition count will be rounded to the closed value if the división is not an integer number

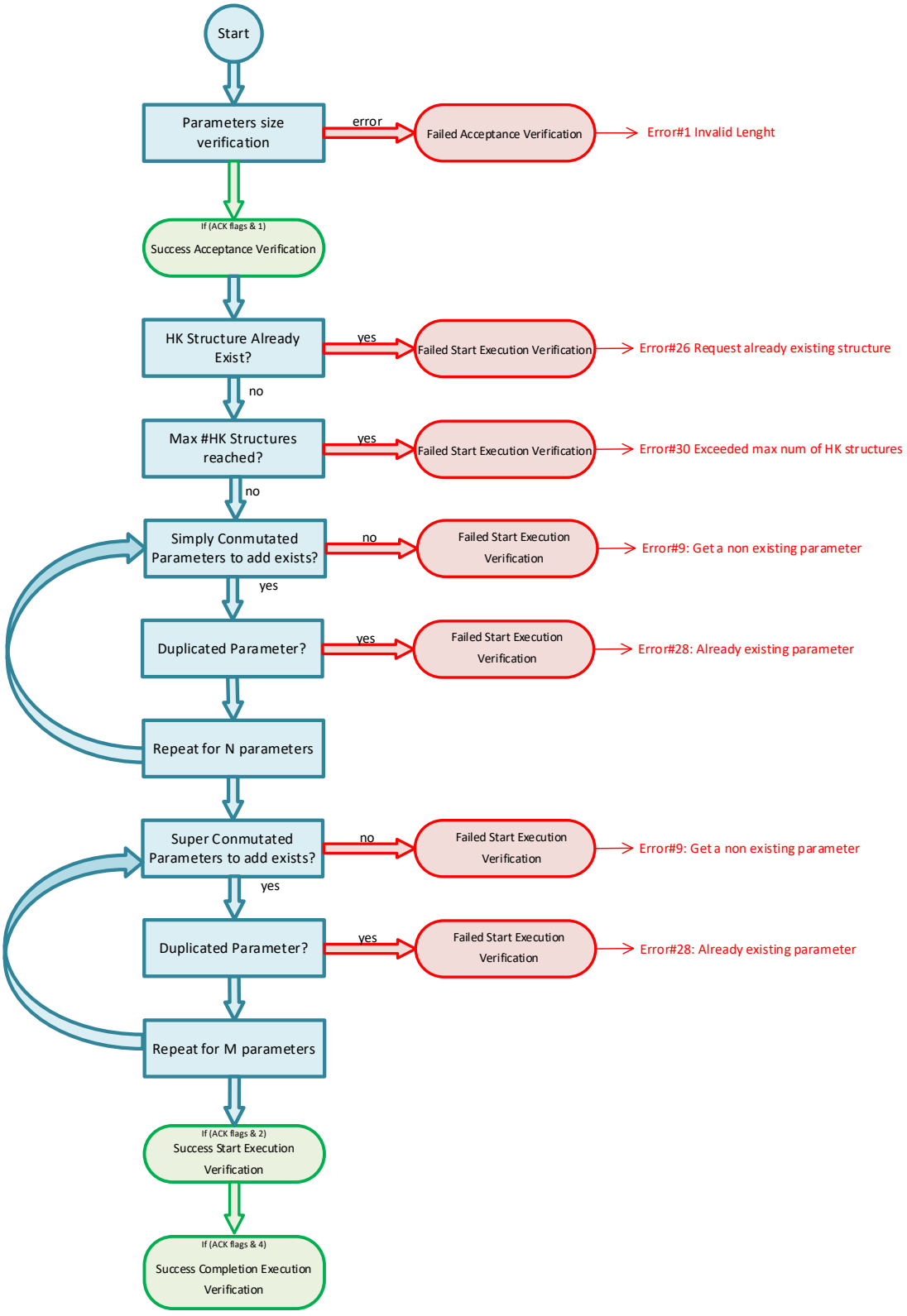
Subtypes requests

TC[3,1] create a housekeeping parameter report structure

Create a housekeeping parameter report structure

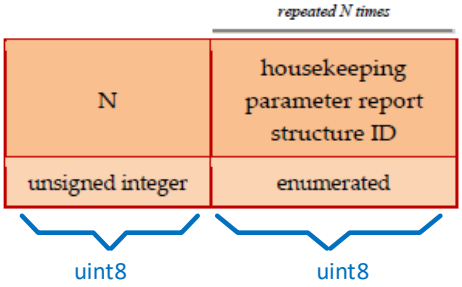


Message request verification flow

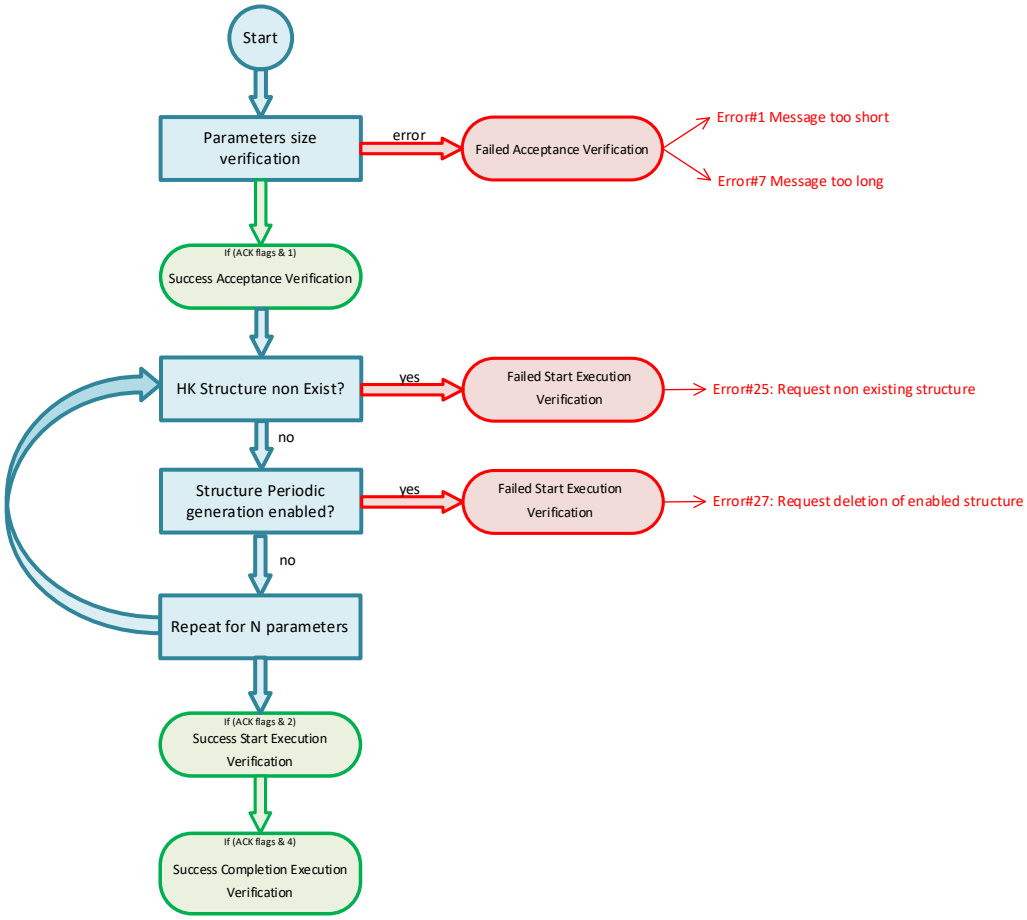


TC[3,3] delete housekeeping parameter report structures

Delete housekeeping parameter report structures

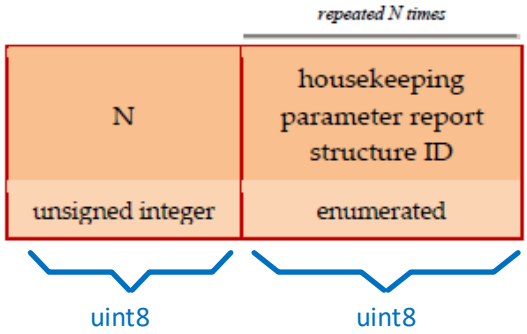


Message request verification flow

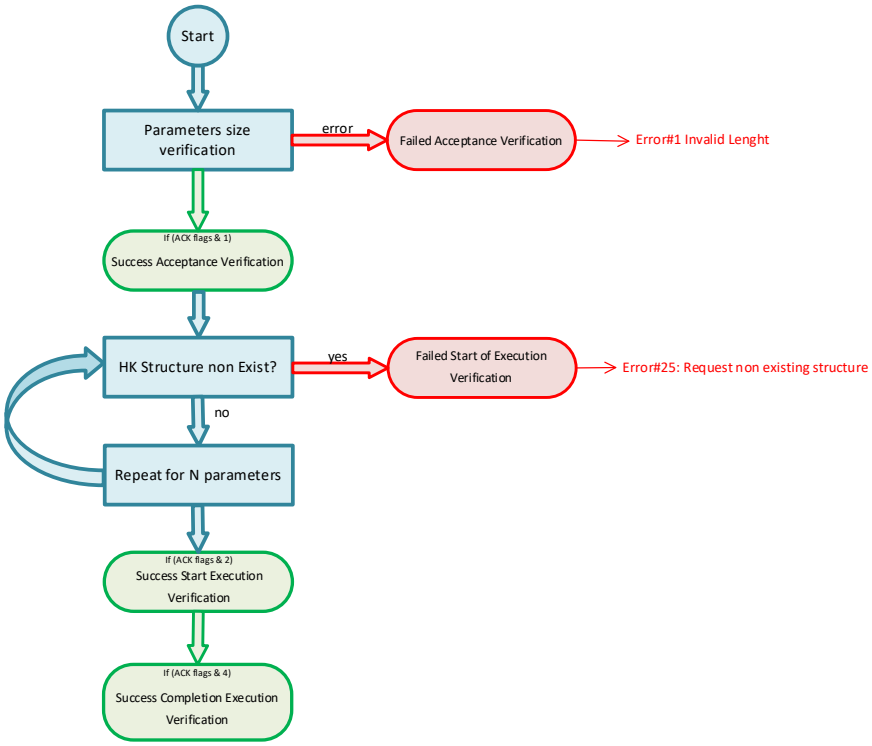


TC[3,5] enable the periodic generation of housekeeping parameter reports

Enable the periodic generation of housekeeping parameter reports

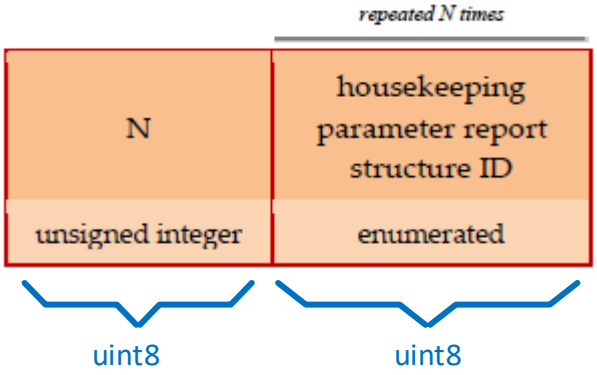


Message request verification flow

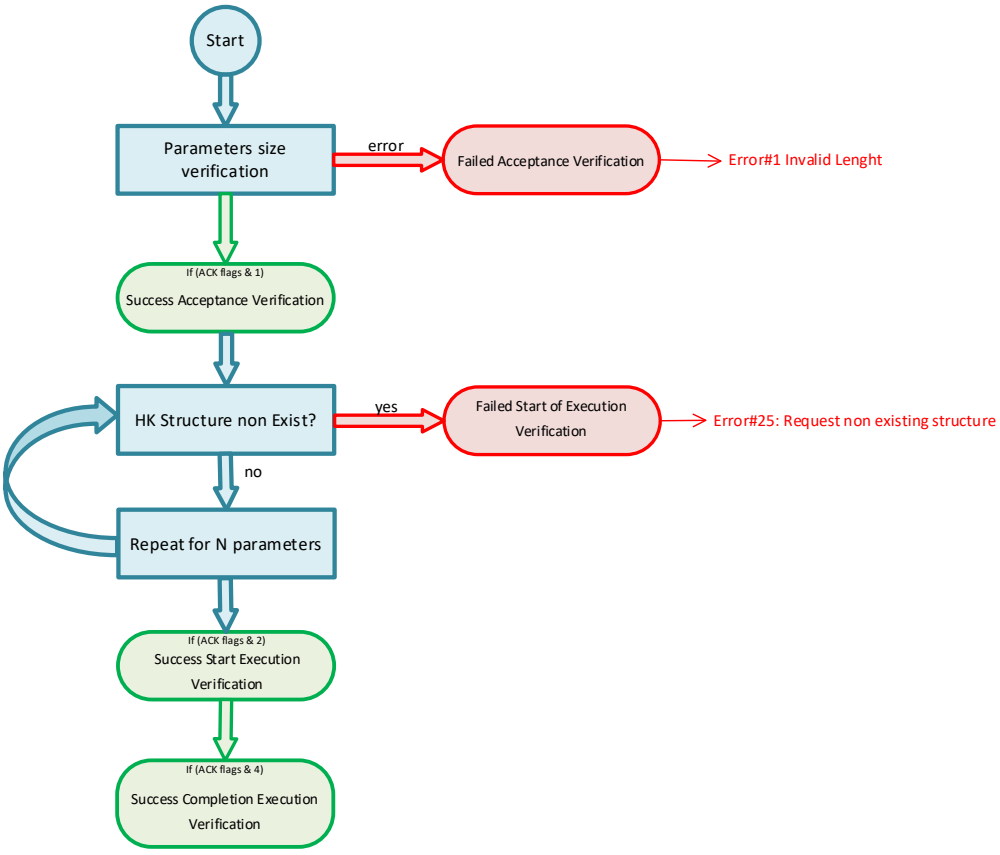


TC[3,6] disable the periodic generation of housekeeping parameter reports

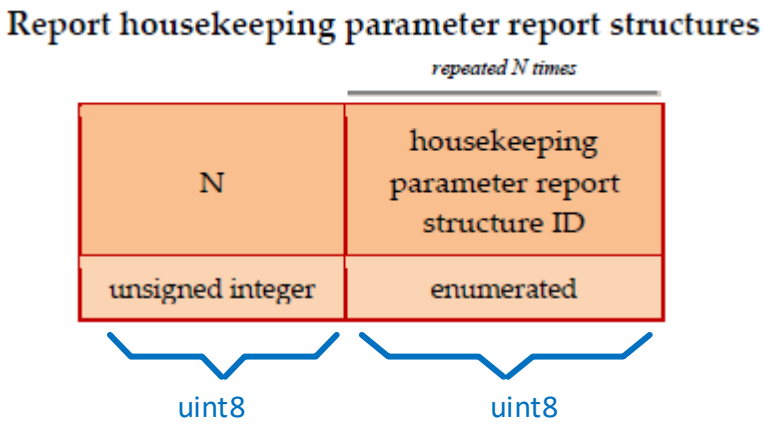
Disable the periodic generation of housekeeping parameter reports



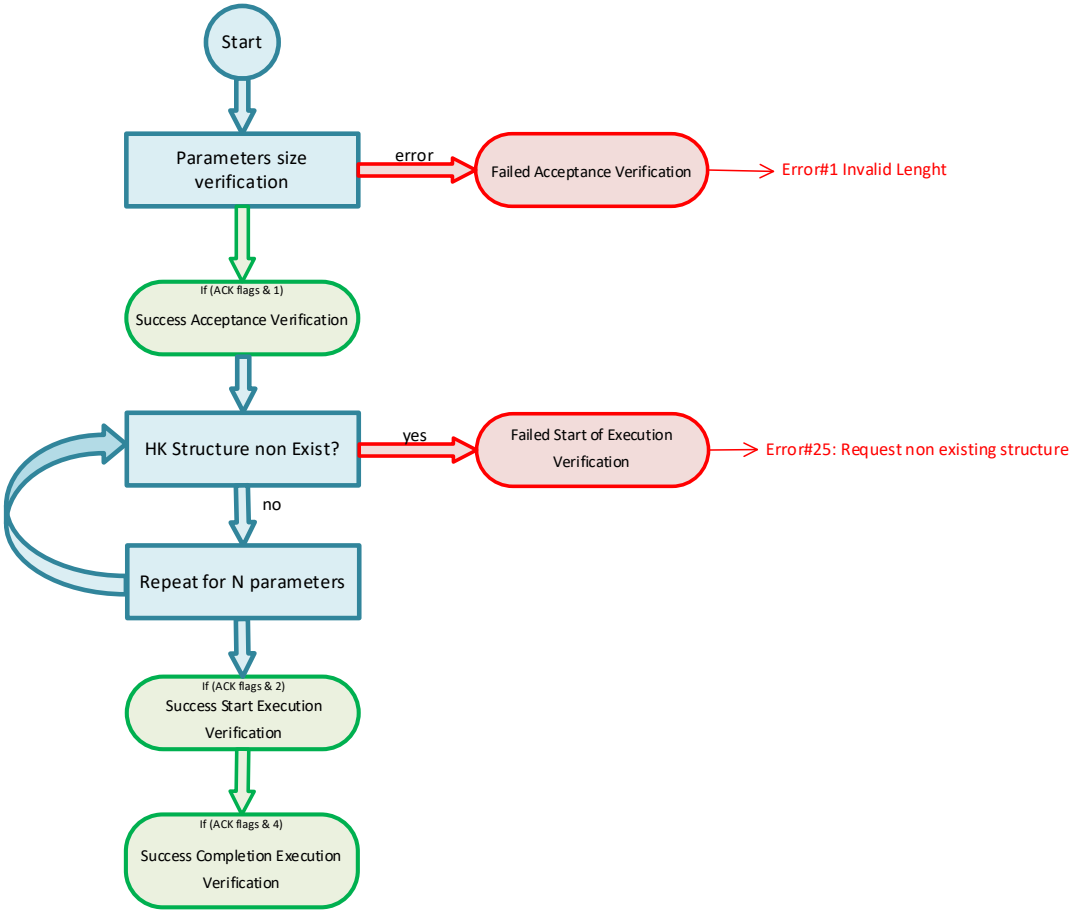
Message request verification flow



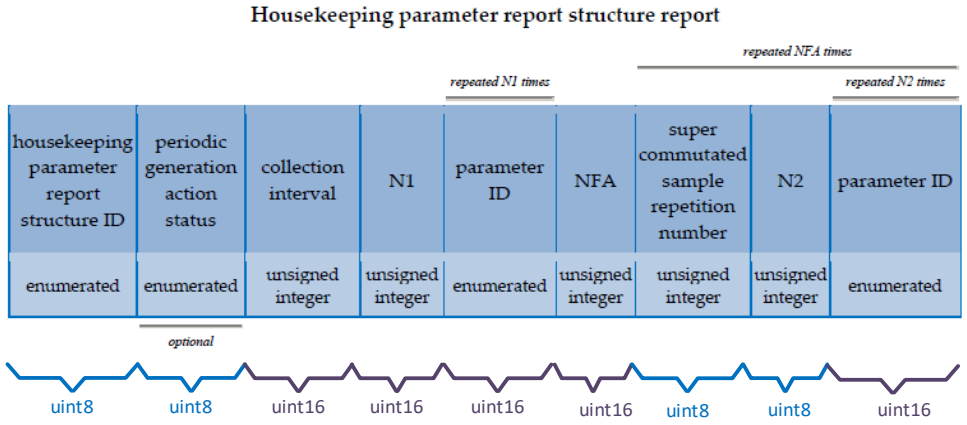
TC[3,9] report housekeeping parameter report structures



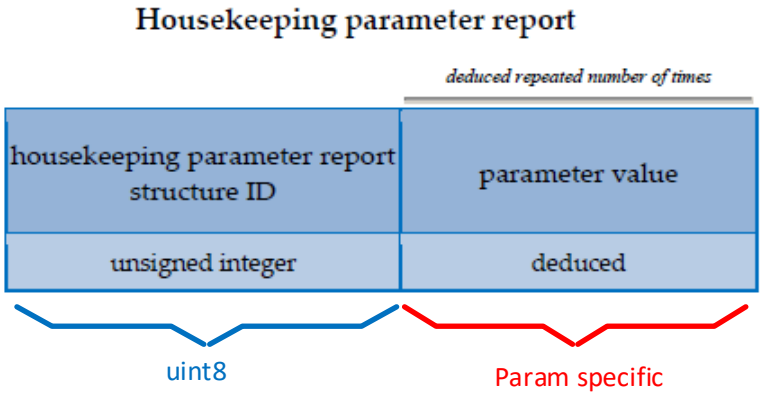
Message request verification flow



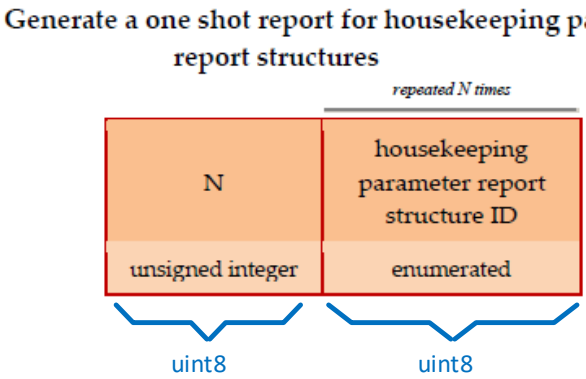
TM[3,10] housekeeping parameter report structure report



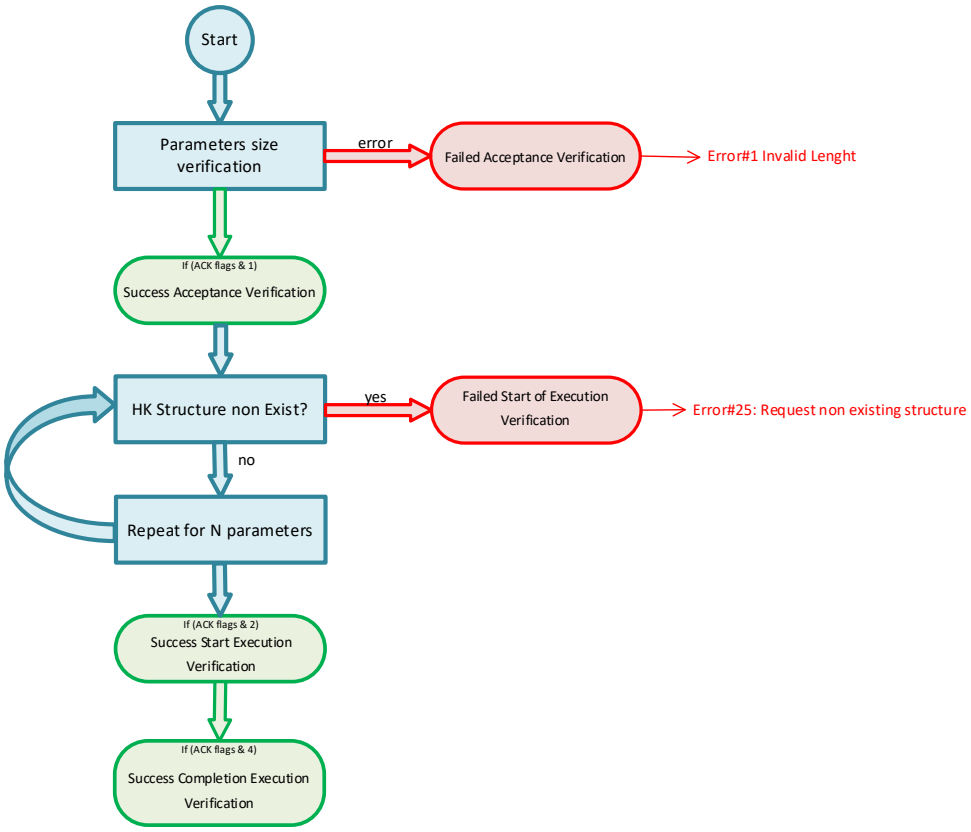
TM[3,25] housekeeping parameter report



TC[3,27] generate a one shot report for housekeeping parameter report structures



Message request verification flow

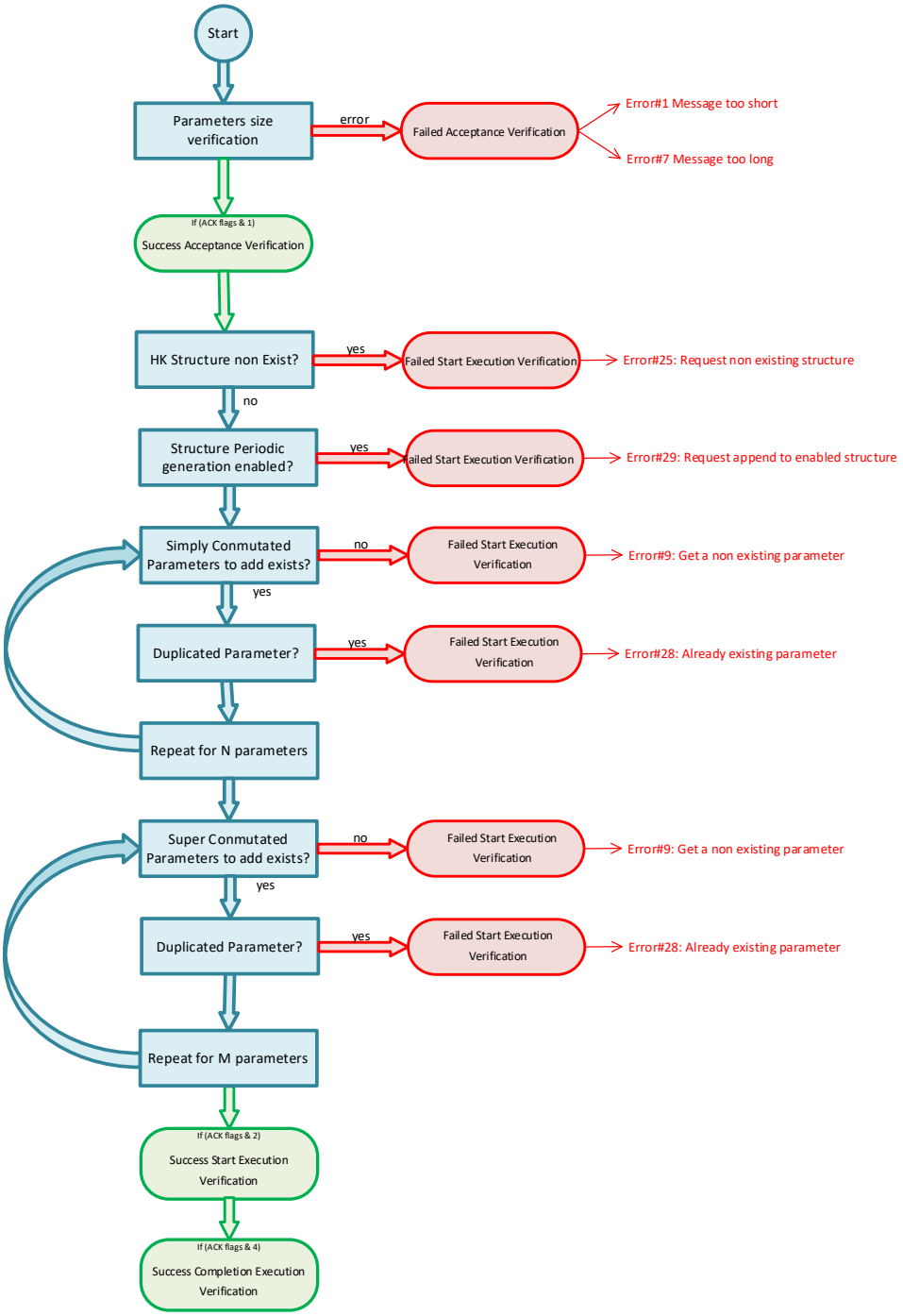


TC[3,29] append parameters to a housekeeping parameter report structure

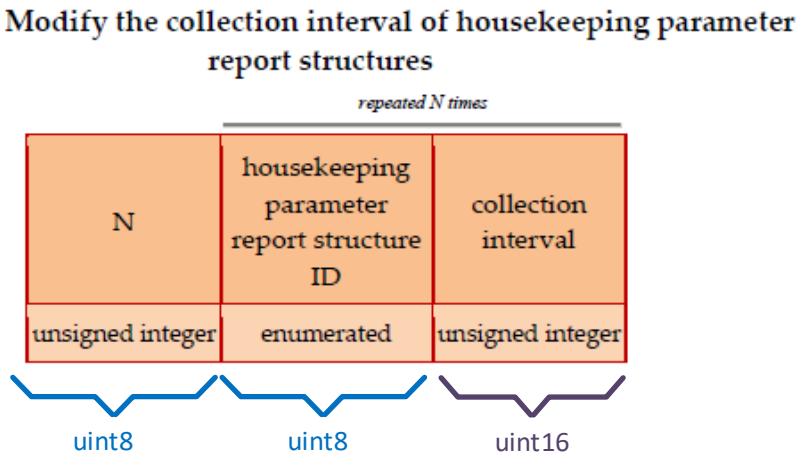
Append parameters to a housekeeping parameter report structure

				repeated N1 times		*repeated NFA times*		*repeated N2 times*
housekeeping parameter report structure ID	N1	parameter ID	NFA	super commutated sample repetition number	N2	parameter ID
enumerated	unsigned integer	enumerated	unsigned integer	unsigned integer	unsigned integer	enumerated
uint8	uint16	uint16	uint16	uint8	uint8	uint16

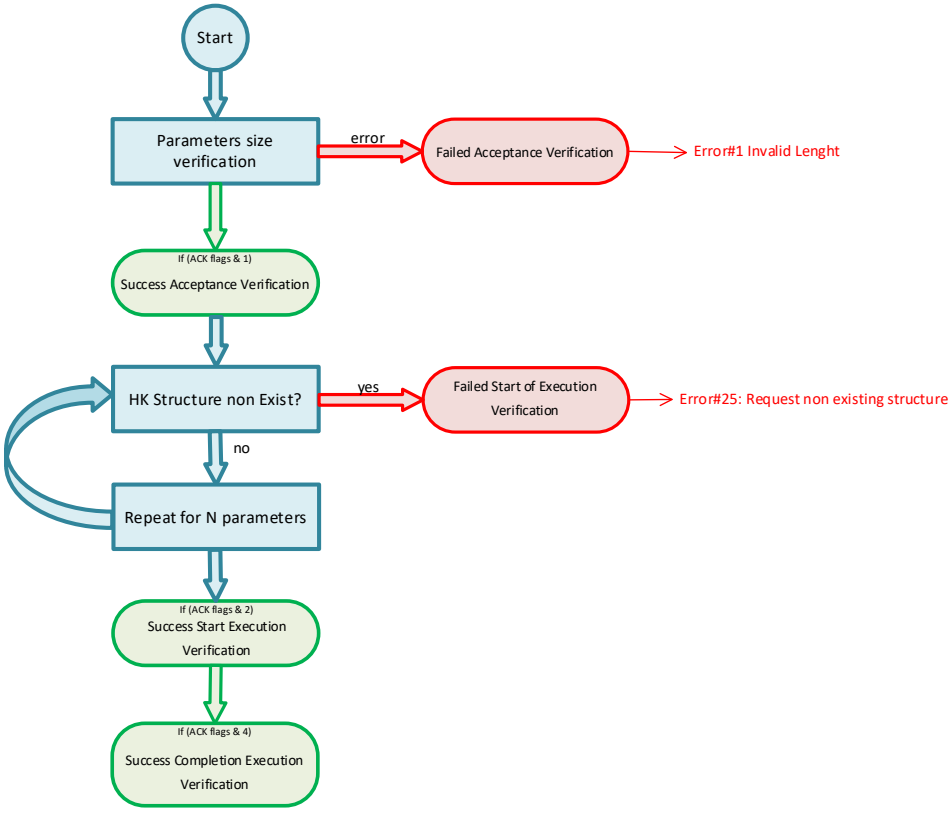
Message request verification flow



TC[3,31] modify the collection interval of housekeeping parameter report structures

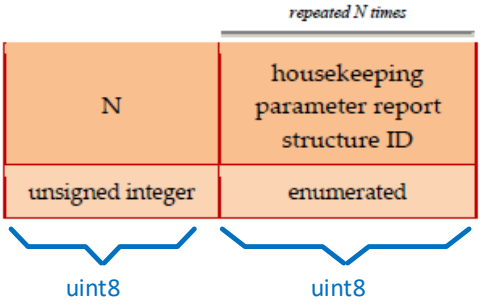


Message request verification flow

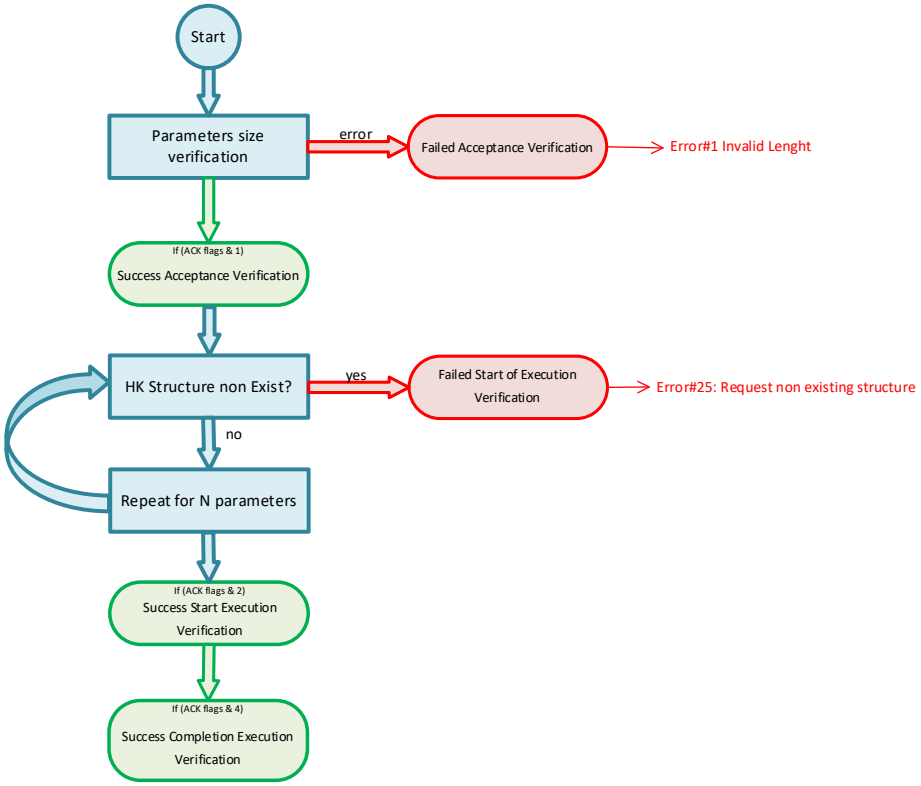


TC[3,34] report the periodic generation properties of diagnostic parameter report structures

Report the periodic generation properties of housekeeping parameter report structures

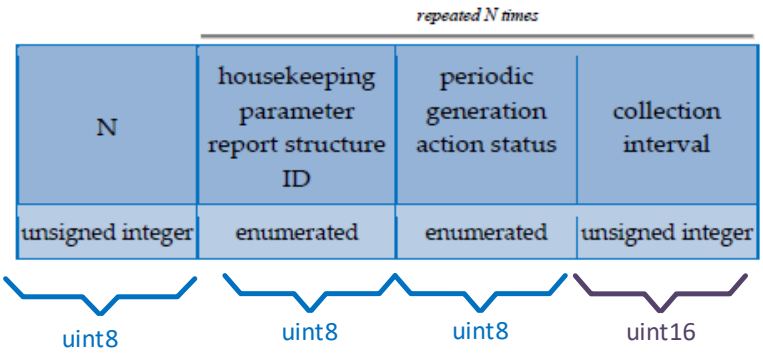


Message request verification flow



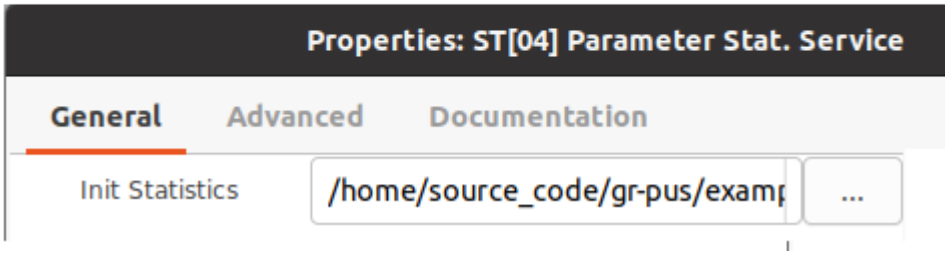
TM[3,35] housekeeping parameter report periodic generation properties report

Housekeeping parameter report periodic generation properties report



2.8 ST[04] PARAMETER STATISTICS

The ***ST[04] Parameter Statistics Service*** block will received all message request at its input port and if those request are for service ST[04] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Init Statistics

Path to the json file with the start up statistics monitoring definitions

Messages

In

The message requests input

Out

The message report output

ver

The message verification output, if an error is detected, an output message for the Request Verification Service ST[01] will be addressed

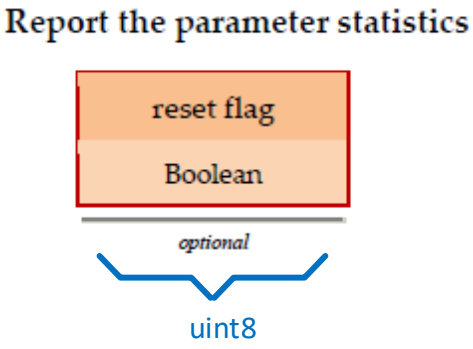
The json init file has next format

```
{
    "periodicEnabled" : true,
    "statistics": [
        {
            "id": 1,
            "interval": 10
        },
        {
            "id": 9,
            "interval": 5
        }
    ]
}
```

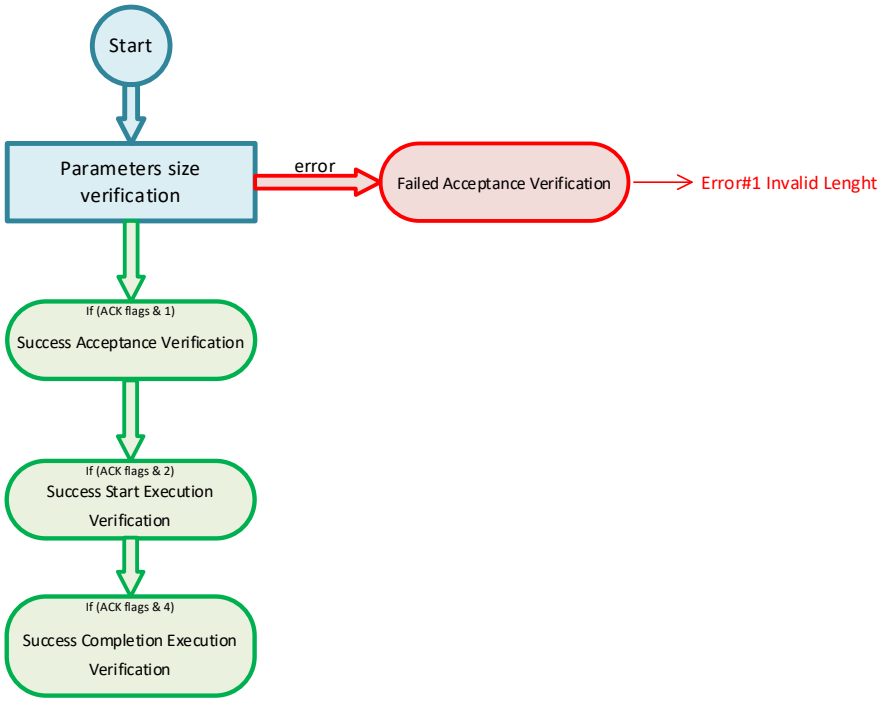
The interval field is expresed in timer tick values, then if timer tick is each 100ms, then “interval” = 10 is 1 sec.

Subtypes requests

TC[04,1] report the parameter statistics



Message request verification flow



TM[04,2] parameter statistics report

Parameter statistics report

repeated N times

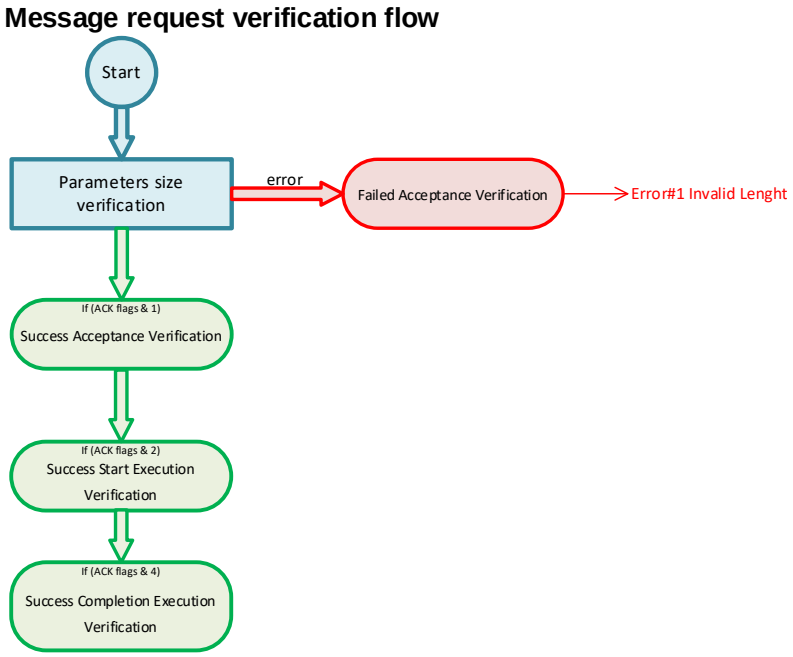
start time	end time	N	parameter ID	number of samples	maximum		minimum		mean value	standard deviation value
					value	time	value	time		
absolute time	absolute time	unsigned integer	enumerated	unsigned integer	deduced	absolute time	deduced	absolute time	deduced	deduced

optional

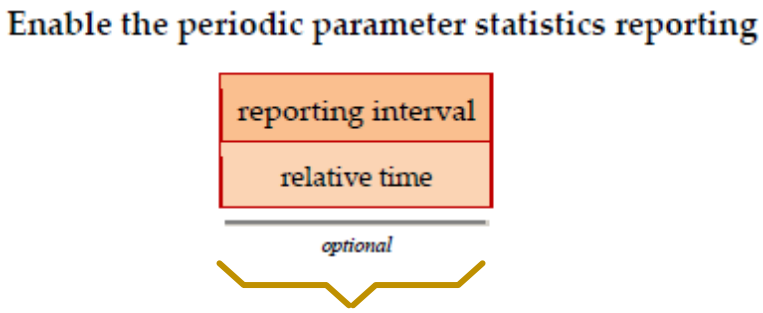
Mission Absolute Time Format uint32_t (*) | Mission Absolute Time Format uint32_t (*) | uint16 | uint16 | uint16 | Param specific | Mission Absolute Time Format uint32_t (*) | Param specific | Mission Absolute Time Format uint32_t (*) | Param specific | Param specific

(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

TC[04,3] reset the parameter statistics



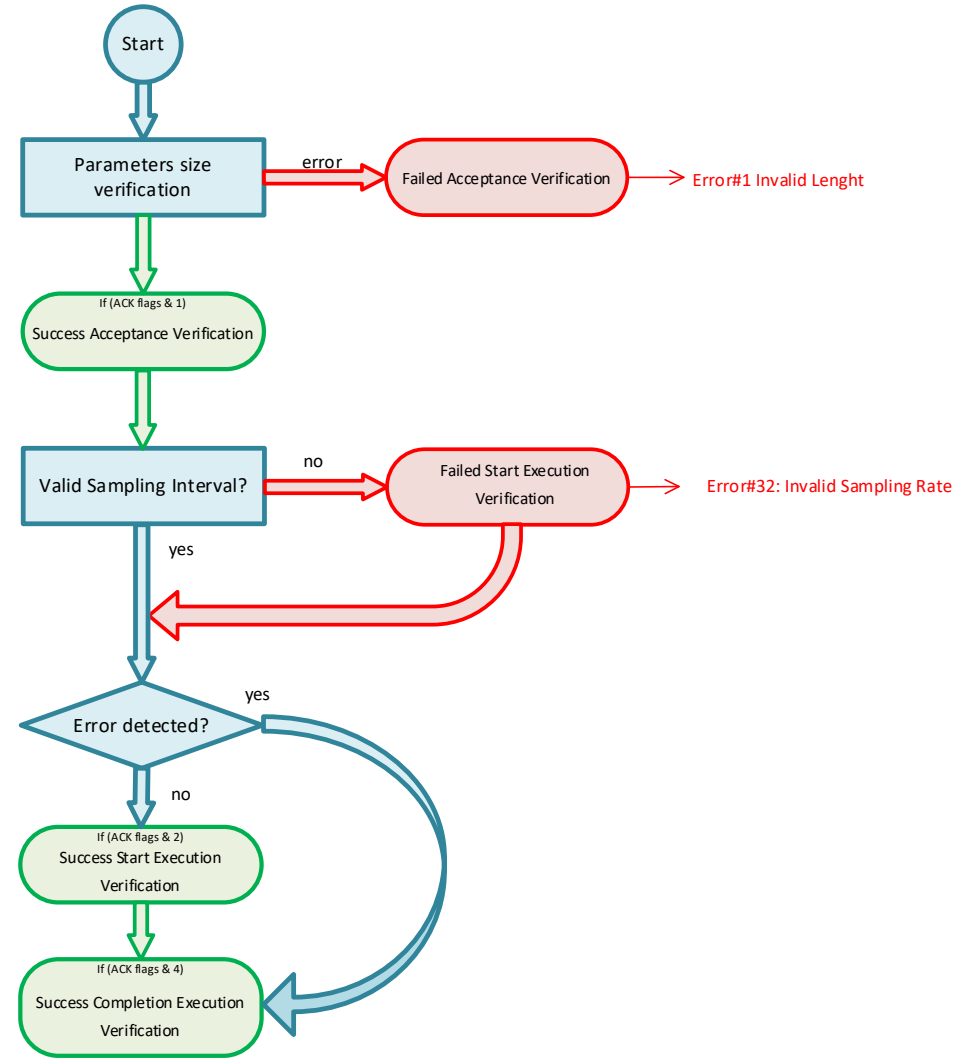
TC[04,4] enable the periodic parameter statistics reporting



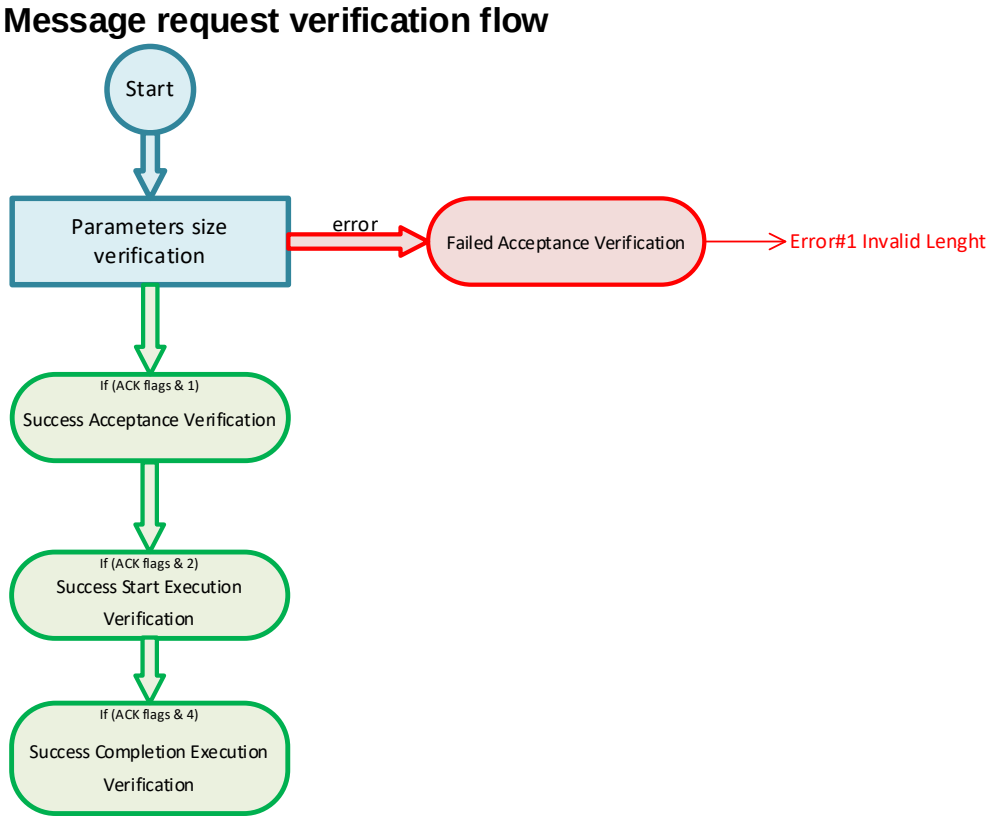
Mission
Relative Time Format
uint32_t (*)

(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

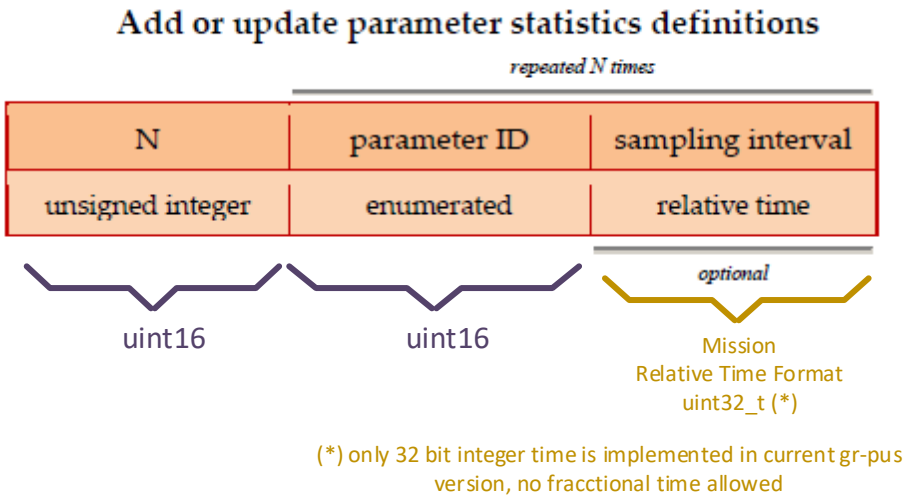
Message request verification flow



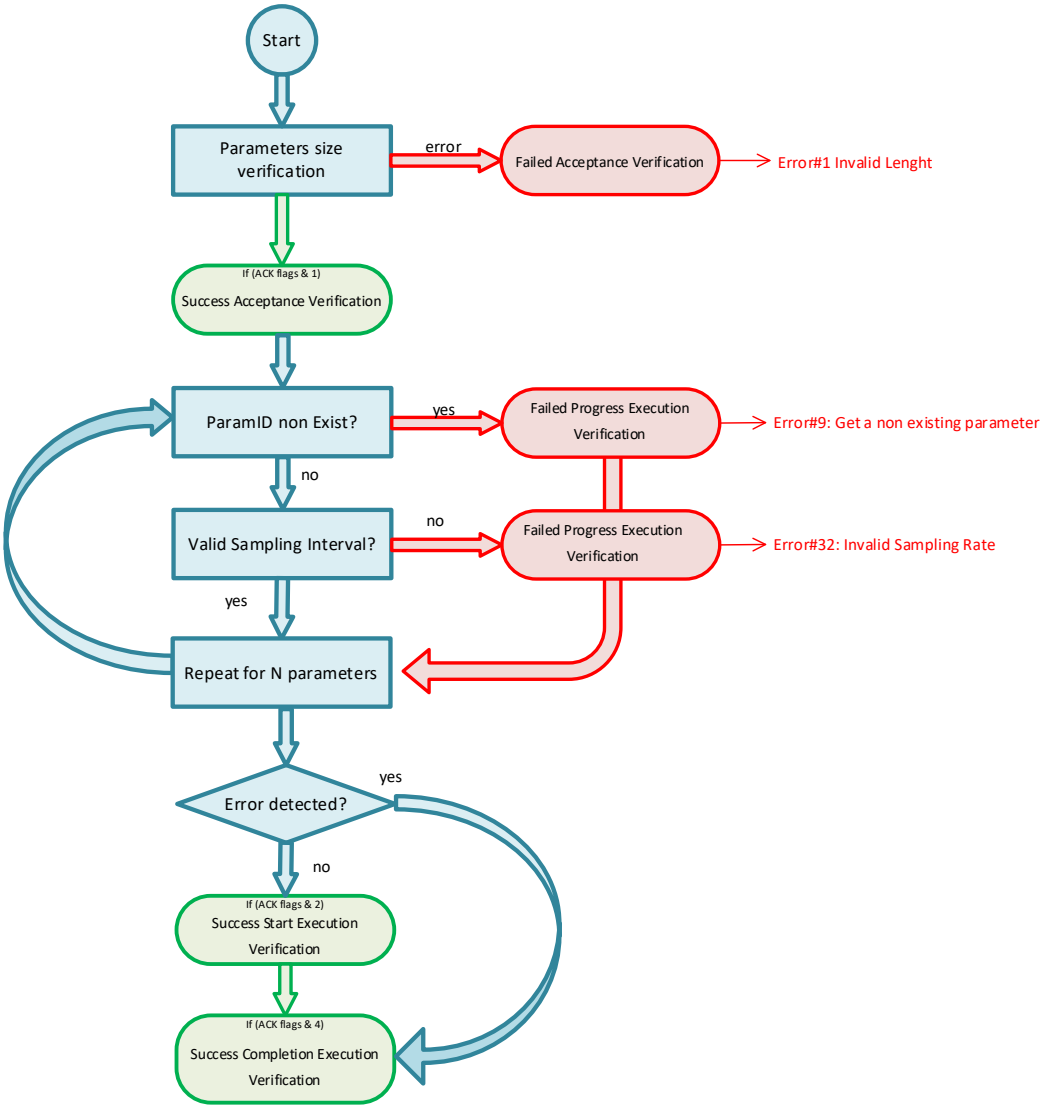
TC[04,5] disable the periodic parameter statistics reporting



TC[04,6] add or update parameter statistics definitions

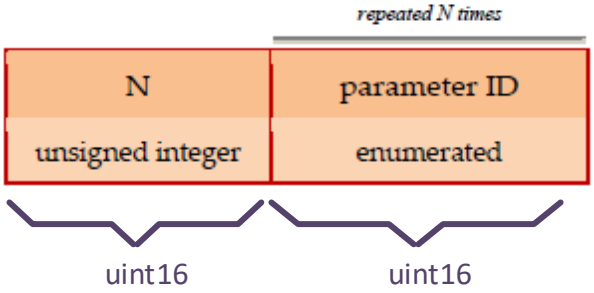


Message request verification flow

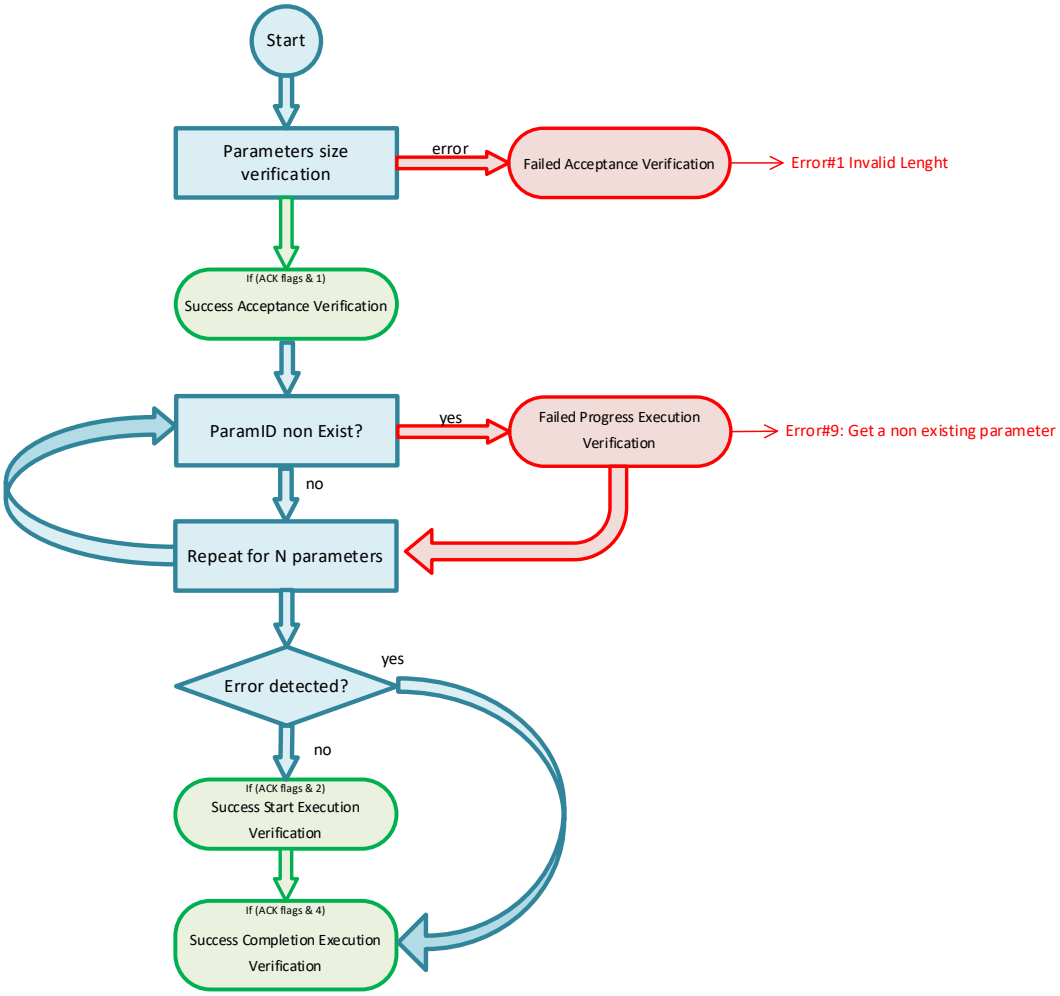


TC[04,7] delete parameter statistics definitions

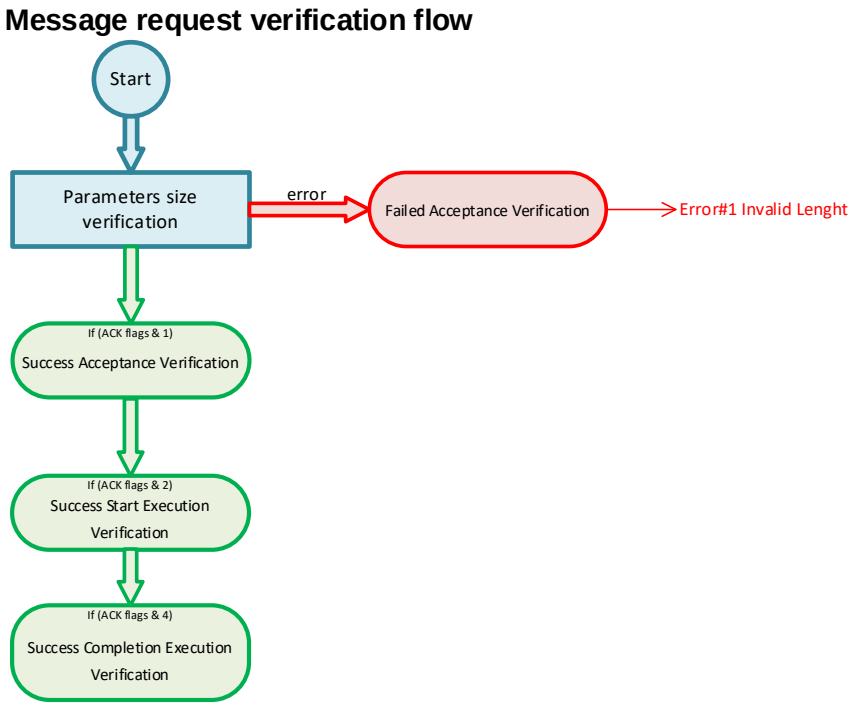
Delete parameter statistics definitions



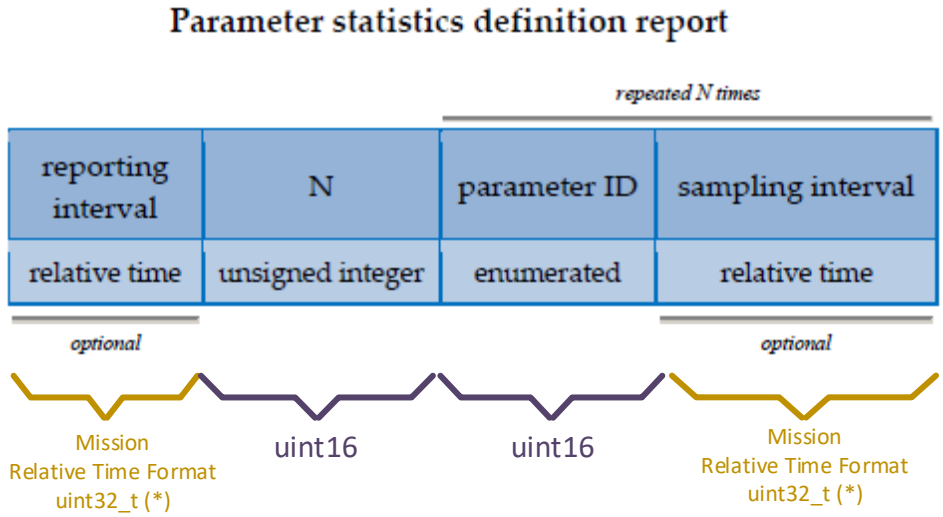
Message request verification flow



TC[04,8] report the parameter statistics definitions



TM[04,9] parameter statistics definition report



(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

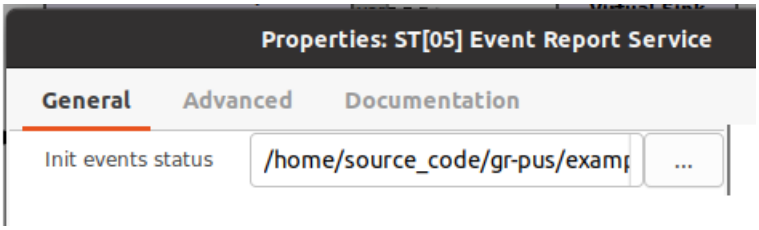
2.9 ST[05] EVENT REPORT

The ***ST[05] Event Report Service*** block will received all message request at its input port and if those request are for service ST[05] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected

The messages with events notification are received in its rid input port. This messages are composed by an optional payload that is copy “as is” in the auxiliary data field in the reports (the onboard monitoring service will includes the RID number which causes the event in this field) and a
pmt::intern(“event”) , pmt::from_long(eventType)

The event types are:

Event Code	Event Description
InformativeUnknownEvent = 0	An unknown event occured
LowSeverityUnknownEvent = 4	An unknown anomaly of low severity anomalyhas occurred
MediumSeverityUnknownEvent = 5	An unknown anomaly of medium severity has occurred
HighSeverityUnknownEvent = 6	An unknown anomaly of high severity has occurred



Parameters

(*R*): *Run-time adjustable*

Init Statistics

Path to the json file with the start up statistics monitoring definitions

Messages

In

The message requests input

Rid

The message event input

Out

The message report output

ver

The message verification output, if an error is detected, an output message for the Request Verification Service ST[01] will be addressed

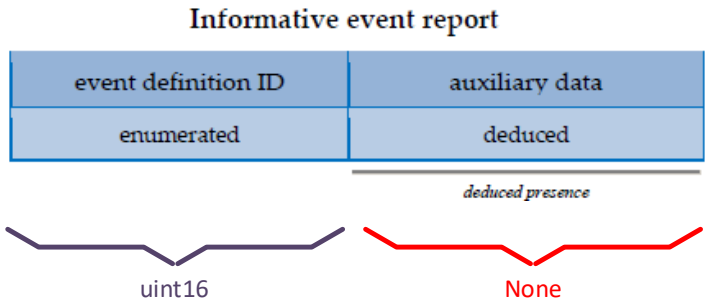
The json init file has next format

```
{
    "events": [
        {
            "id": 0,
            "enabled": true
        },
        {
            "id": 1,
            "enabled": false
        },
        {
            "id": 2,
            "enabled": false
        },
        {
            "id": 3,
            "enabled": false
        },
        {
            "id": 4,
            "enabled": true
        },
        {
            "id": 5,
            "enabled": true
        },
        {
            "id": 6,
            "enabled": true
        }
    ]
}
```

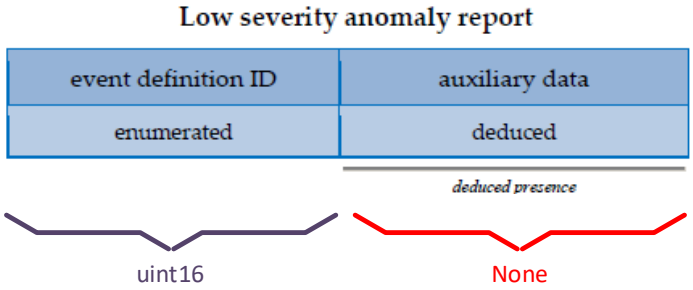
Where the event ID which match the event type could be enabled or disabled at start up

Subtypes requests

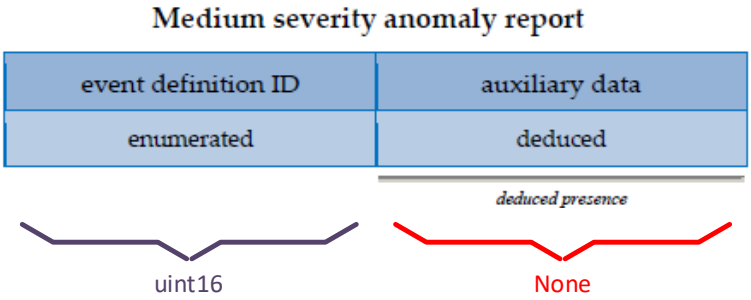
TM[05,1] informative event report



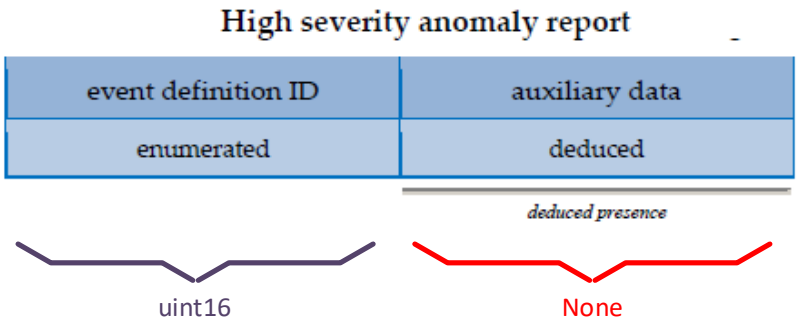
TM[05,2] low severity anomaly report



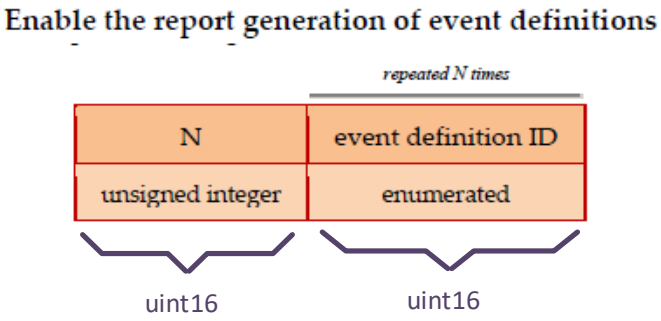
TM[05,3] Medium severity anomaly report



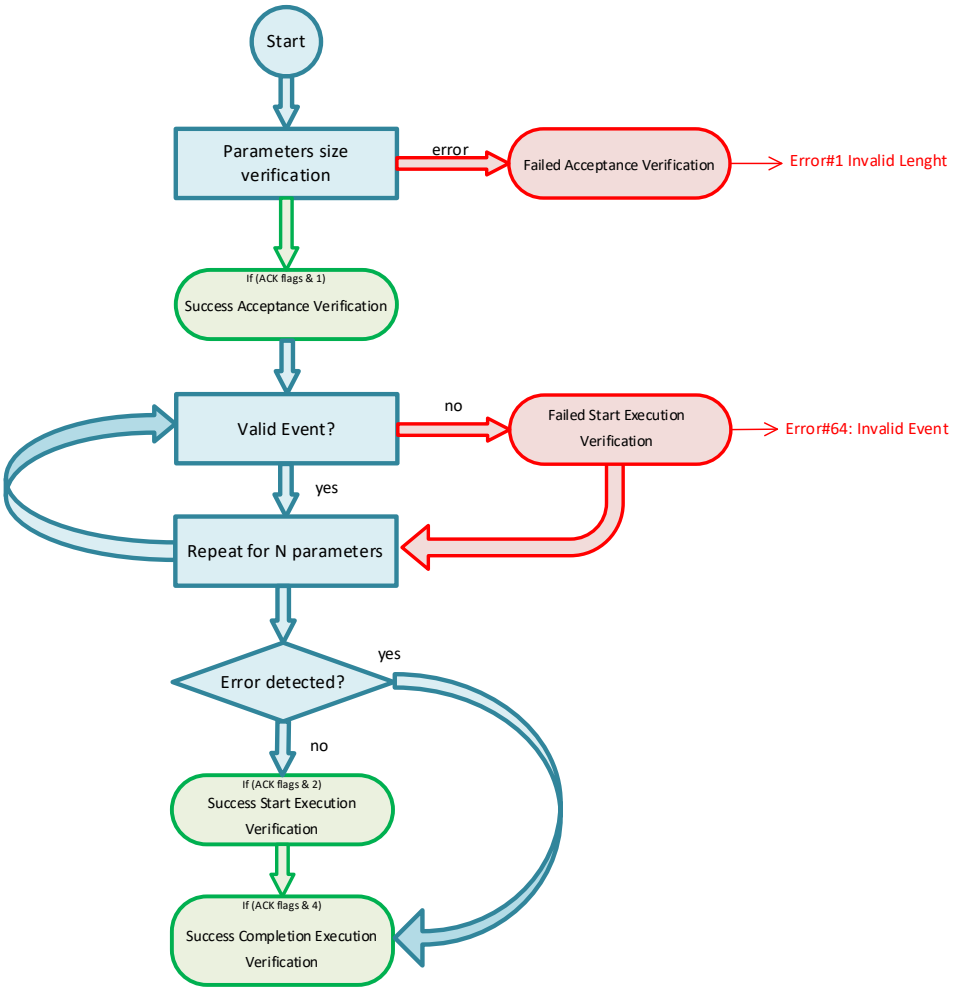
TM[05,4] High severity anomaly report



TC[5,5] enable the report generation of event definitions



Message request verification flow



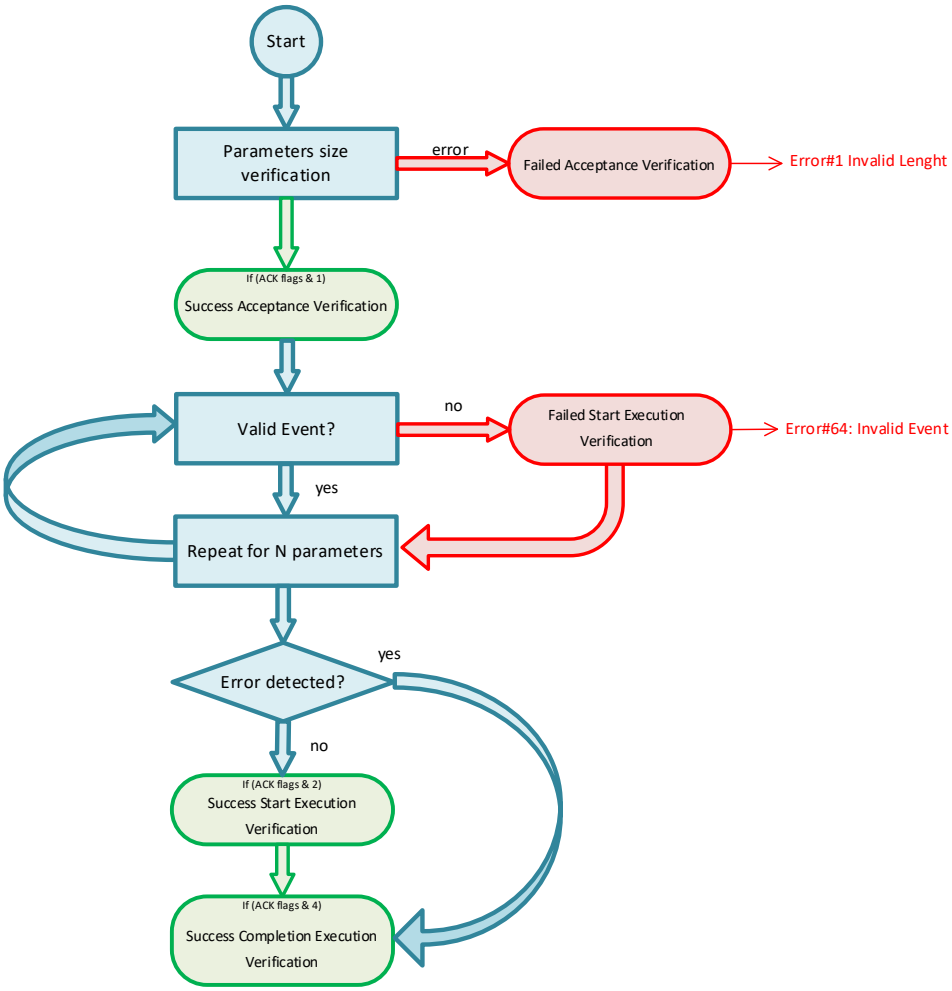
TC[5,6] disable the report generation of event definitions

Disable the report generation of event definitions

repeated N times

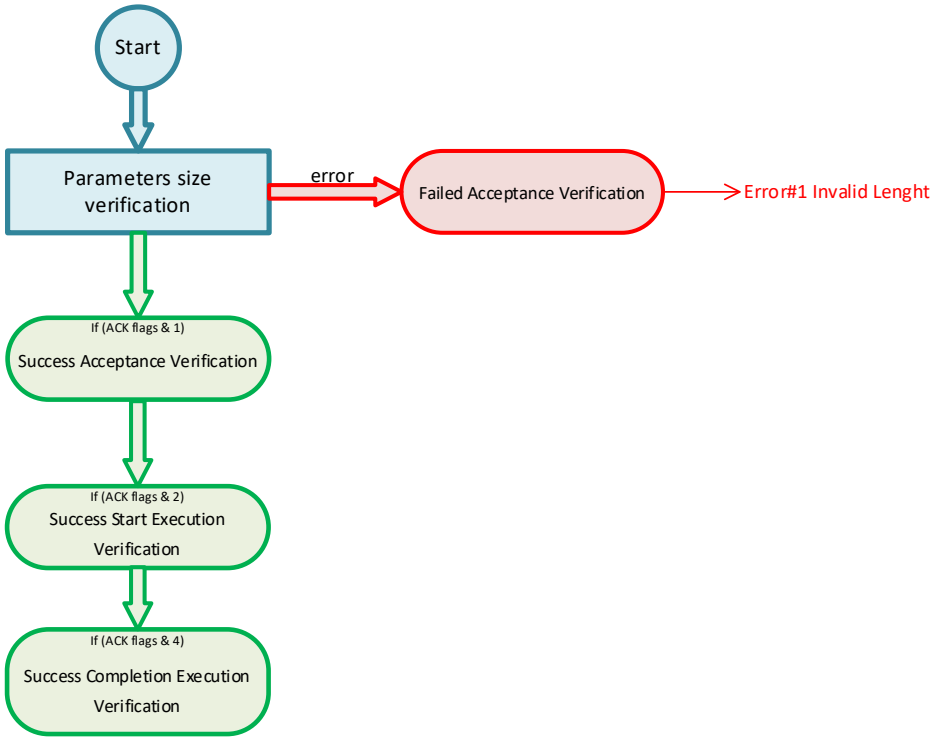
N	event definition ID
unsigned integer	enumerated
uint16	uint16

Message request verification flow

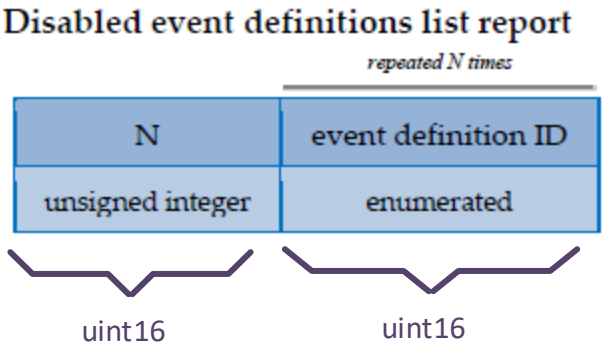


TC[5,7] report the list of disabled event definitions

Message request verification flow



TM[5,8] disabled event definitions list report



2.10 ST[06] MEMORY MANAGEMENT SERVICE

The ***ST[06] Memory Management Service*** block will received all message request at its input port and if those request are for service ST[06] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected

Note:
This service is hardware dependent, then, it required an applicaiton specific code for each implementation, see and/or the MemoryManager class

Parameters

(*R*): *Run-time adjustable*

Messages

In

The message requests input

Out

The message report output

ver

The message verification output, if an error is detected, an output message for the Request Verification Service ST[01] will be addressed

Subtypes requests

TC[6,2] load raw memory data areas

Load raw memory data areas

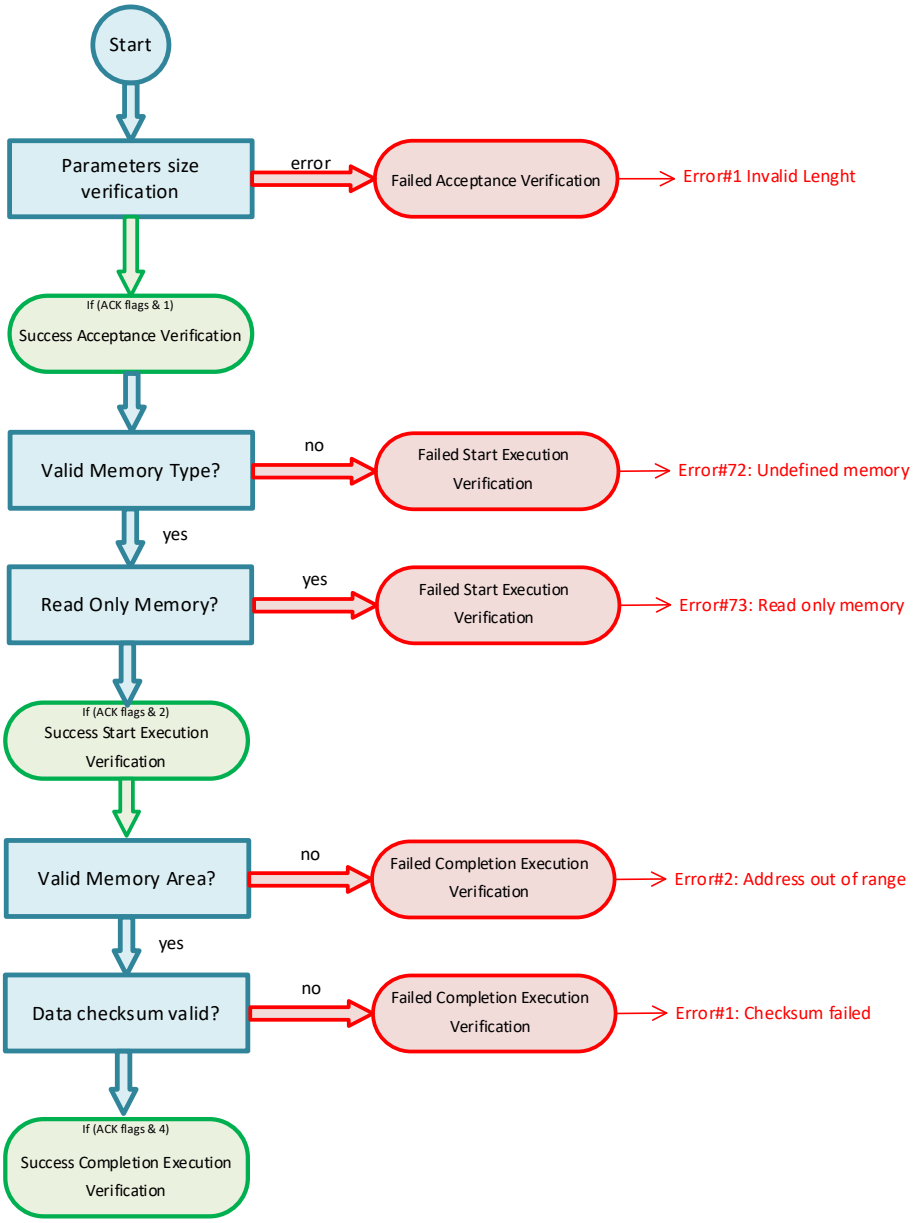
repeated N times

memory ID	N	start address	data to load		checksum
			length	data	
enumerated	unsigned integer	unsigned integer	variable octet-string		bit-string (16 bits)

optional *optional*

NOTE The PFC of the length field of the data to load is driven by requirement 7.3.8d.

Message request verification flow



TC[6,5] dump raw memory data

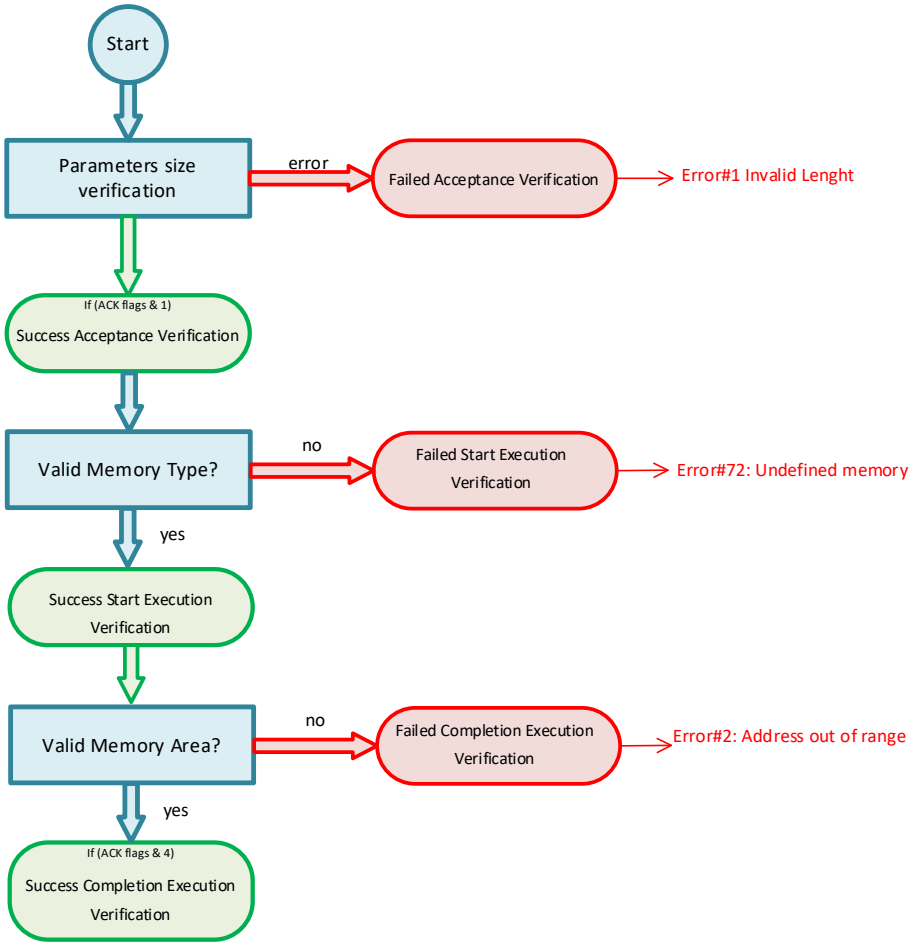
Dump raw memory data

repeated N times

memory ID	N	start address	length
enumerated	unsigned integer	unsigned integer	unsigned integer

optional

Message request verification flow



TM[6,6] dumped raw memory data report

Dumped raw memory data report

repeated N times

memory ID	N	start address	dumped data		checksum
			length	data	
enumerated	unsigned integer	unsigned integer	variable octet-string		bit-string (16 bits)

optional *optional*

NOTE The PFC of the length field of the dumped data is driven by requirement 7.3.8d.

TC[6,9] check raw memory data

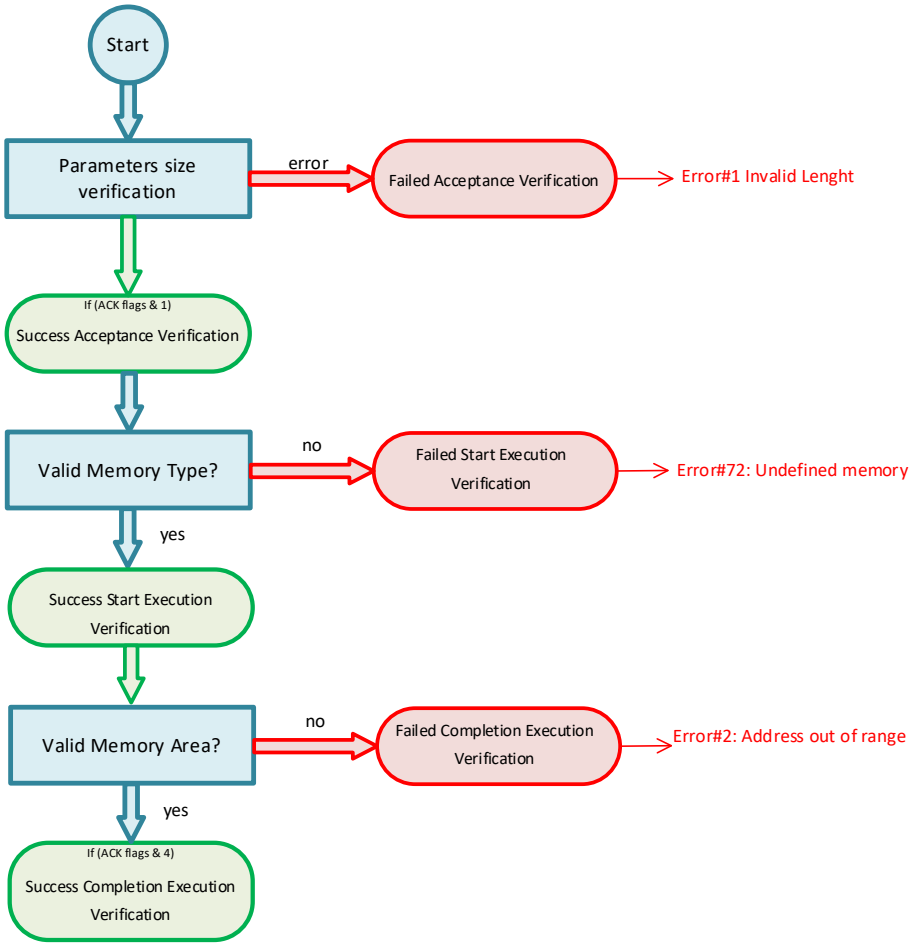
Check raw memory data

repeated N times

memory ID	N	start address	length
enumerated	unsigned integer	unsigned integer	unsigned integer

optional

Message request verification flow



TM[6,10] checked raw memory data report

Checked raw memory data report

repeated N times

memory ID	N	start address	length	checksum
enumerated	unsigned integer	unsigned integer	unsigned integer	bit-string (16 bits)

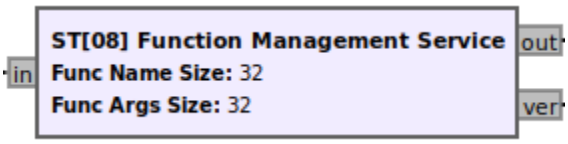
optional

uint8 uint16 uint64 uint16 uint16

2.11 ST[08] FUNCTION MANAGEMENT SERVICE

The ***ST[08] Function Management Service*** block will received all message request at its input port and if those request are for service ST[08] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected

Note:
This service is application dependent, then, it required an specific code for each application, see and/or modify the functionInit class



Parameters

(*R*): *Run-time adjustable*

Func Name Size

The function name uses fixed string, then this value defines the character size of the function names

Func Args Size

The function arguments bytes fixed size for all functions

Messages

In

The message requests input

Out

The message report output

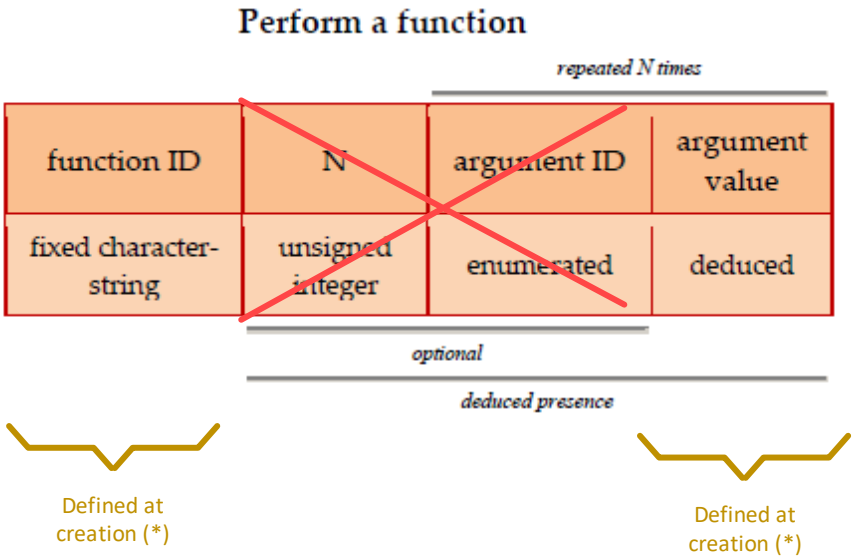
ver

The message verification output, if an error is detected, an output message for the Request Verification Service ST[01] will be addressed

Functions implementation are mission specific, check out the ***FunctionInit*** block used for the unitary test as example on how to link a mission function with this service

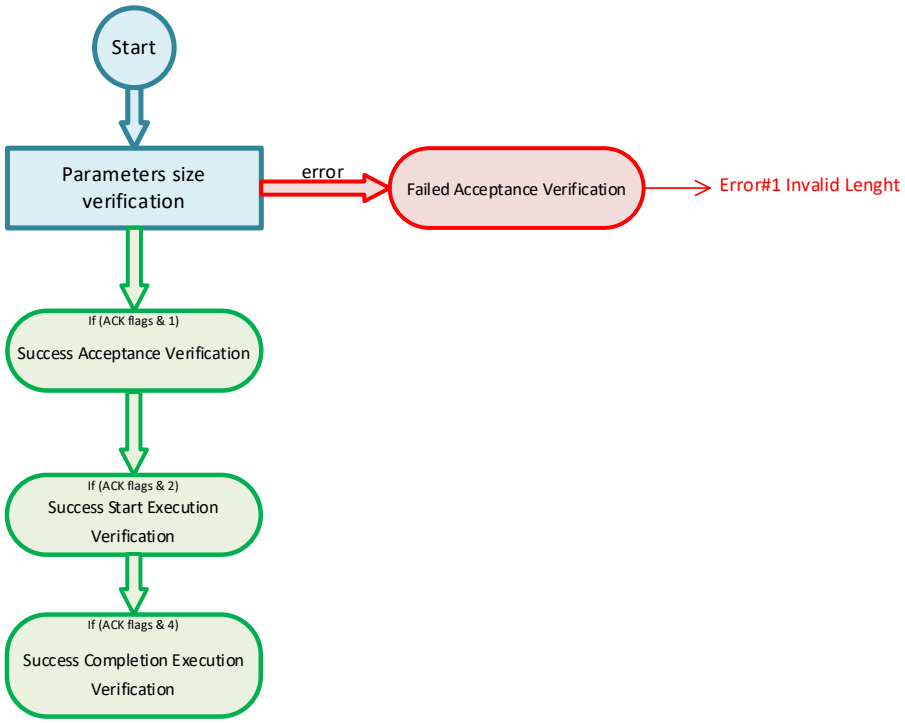
Subtypes requests

TC[8,1] perform a function



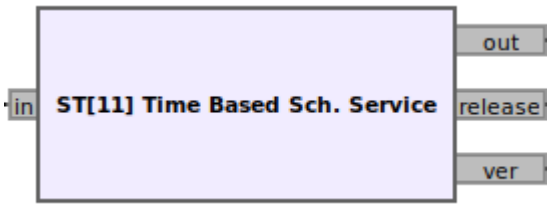
(*) function ID size and Argument size is defined upon block creation for each mission

Message request verification flow



2.12 ST[11] TIME BASED SCHEDULING SERVICE

The ***ST[11] Time Based Scheduling Service*** block will received all message request at its input port and if those request are for service ST[11] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Messages

In

The message requests input

Out

The message report output

ver

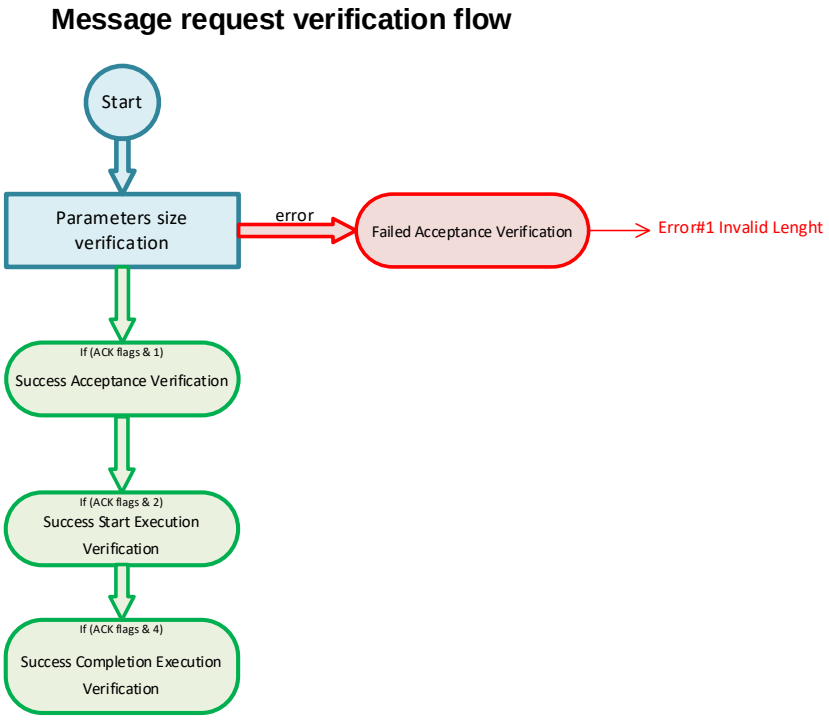
The message verification output, if an error is detected, an output message for the Request Verification Service ST[01] will be addressed

Release

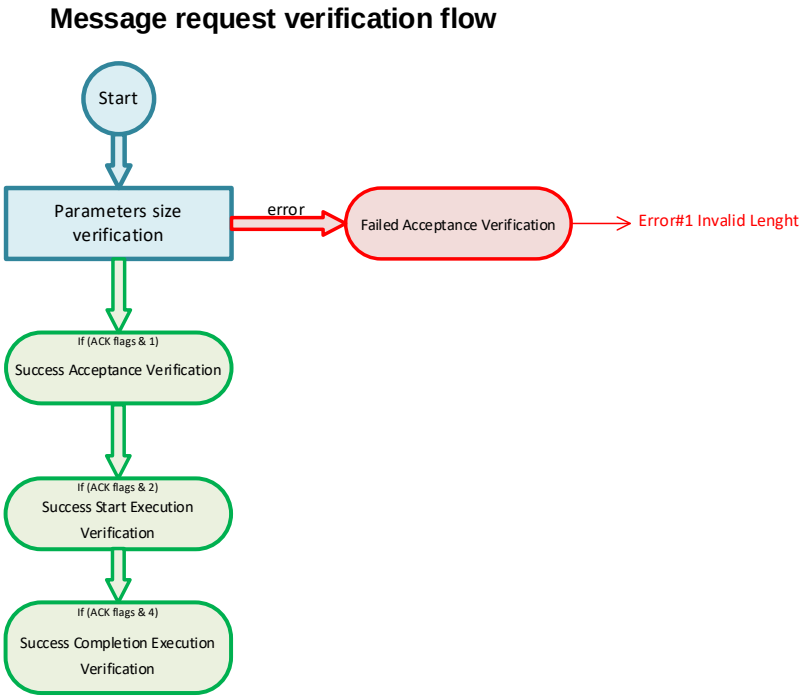
The schedule messages release output, each time a message is released from scheduler, the message will be output through this port

Subtypes requests

TC[11,1] enable the time-based schedule execution function

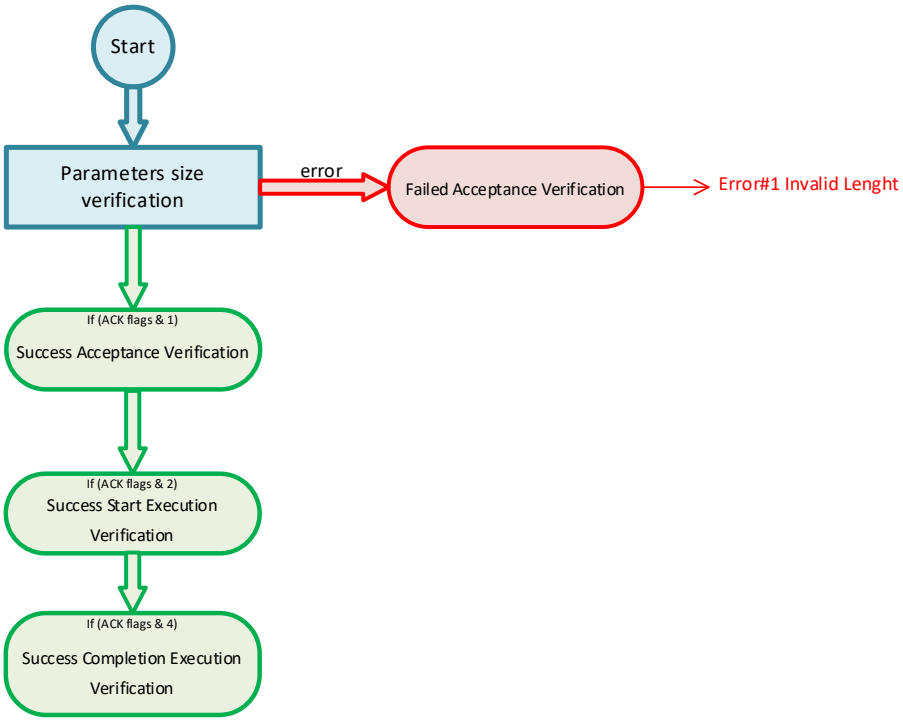


TC[11,2] disable the time-based schedule execution function



TC[11,3] reset the time-based schedule

Message request verification flow



TC[11,4] insert activities into the time-based schedule

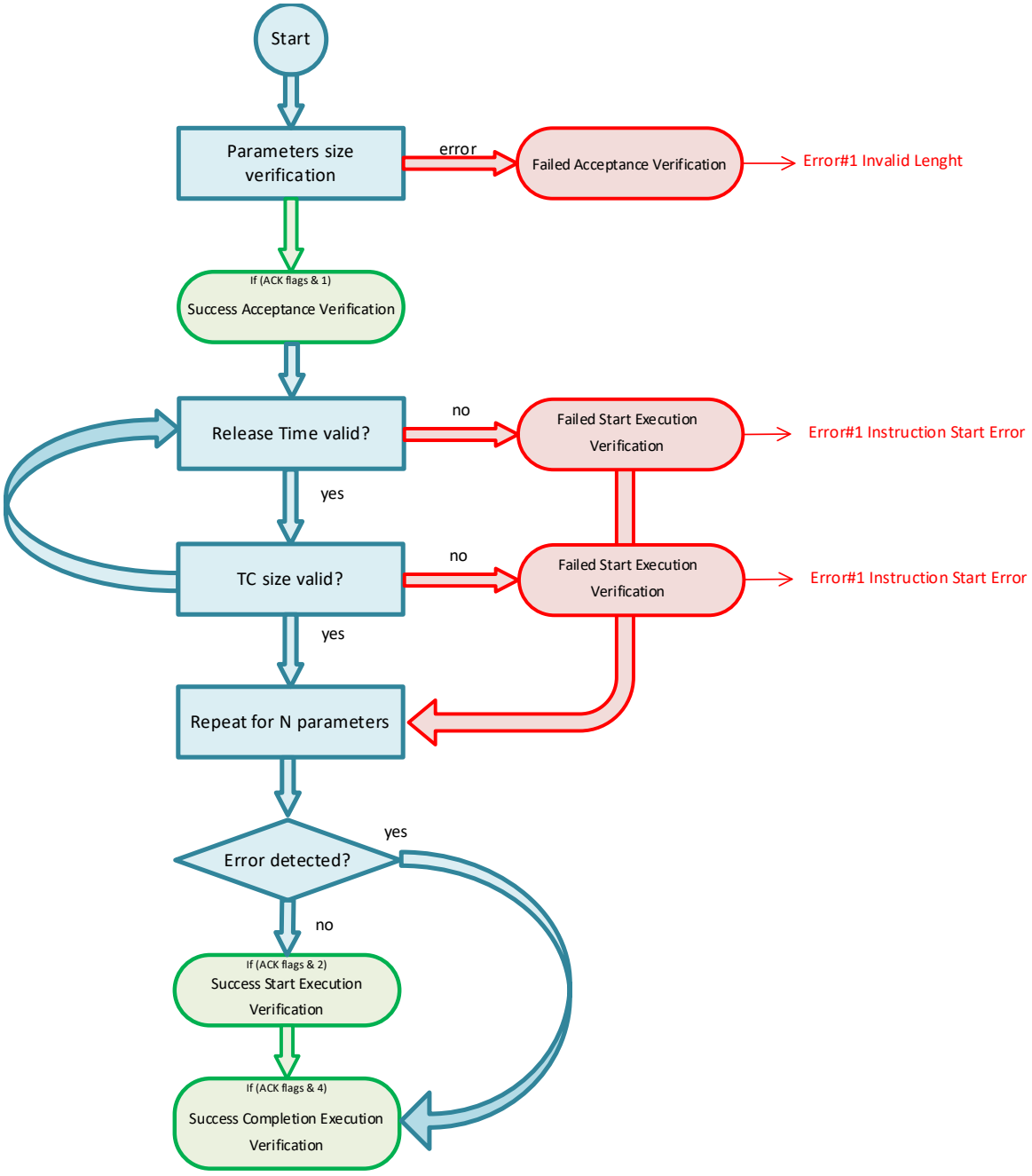
Insert activities into the time-based schedule

repeated N times

sub-schedule ID	N	group ID	release time	request
enumerated	unsigned integer	enumerated	absolute time	TC packet
optional		*optional*		
	uint16		Mission Absolute Time Format uint32_t (*)	Variable

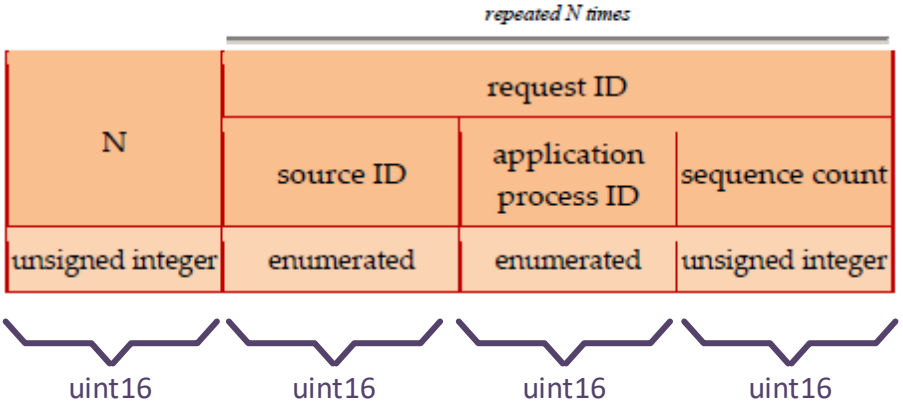
(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

Message request verification flow

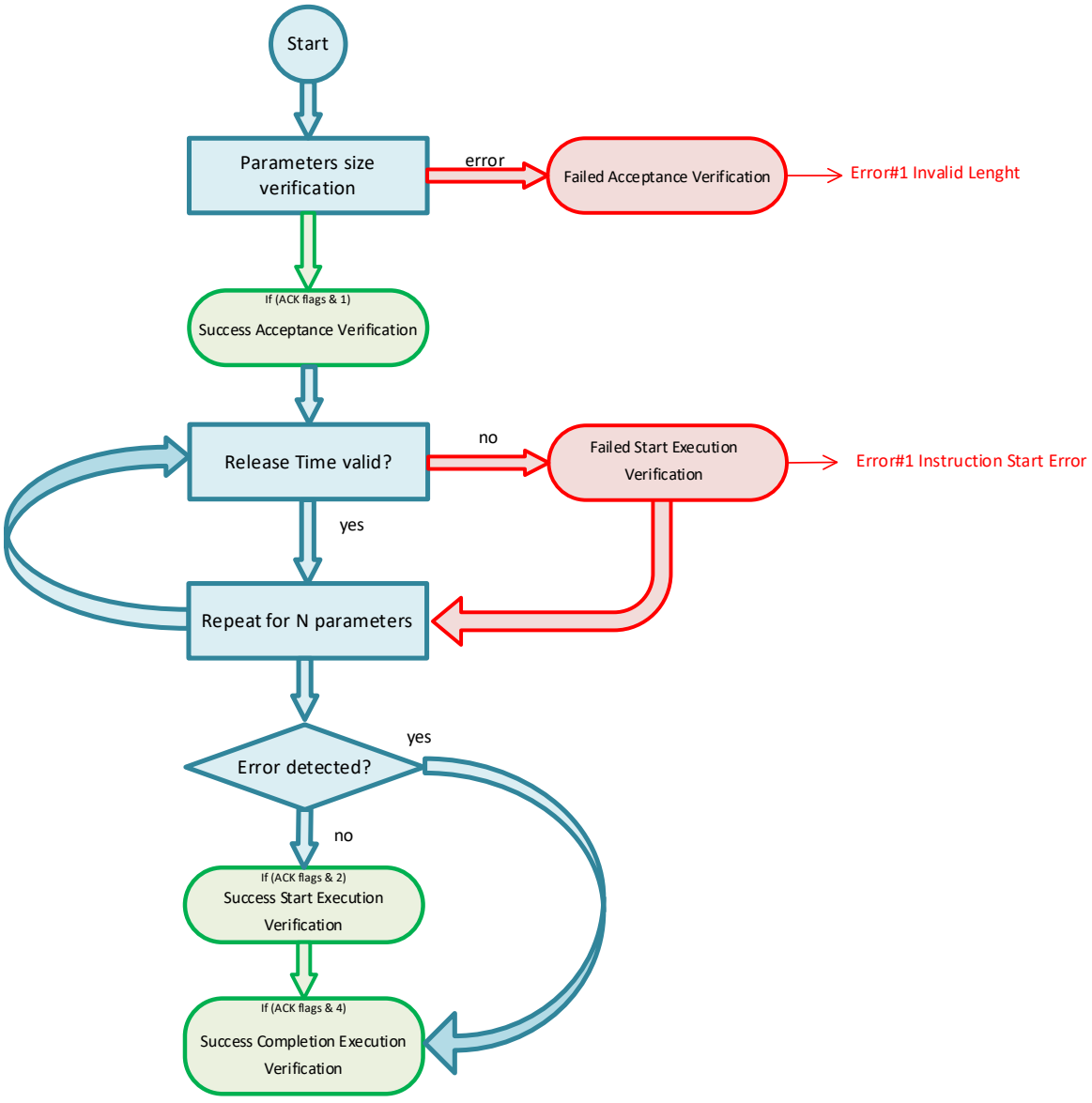


TC[11,5] delete time-based scheduled activities identified by request identifier

Delete time-based scheduled activities identified by request identifier



Message request verification flow



TC[11,7] time-shift scheduled activities identified by request identifier

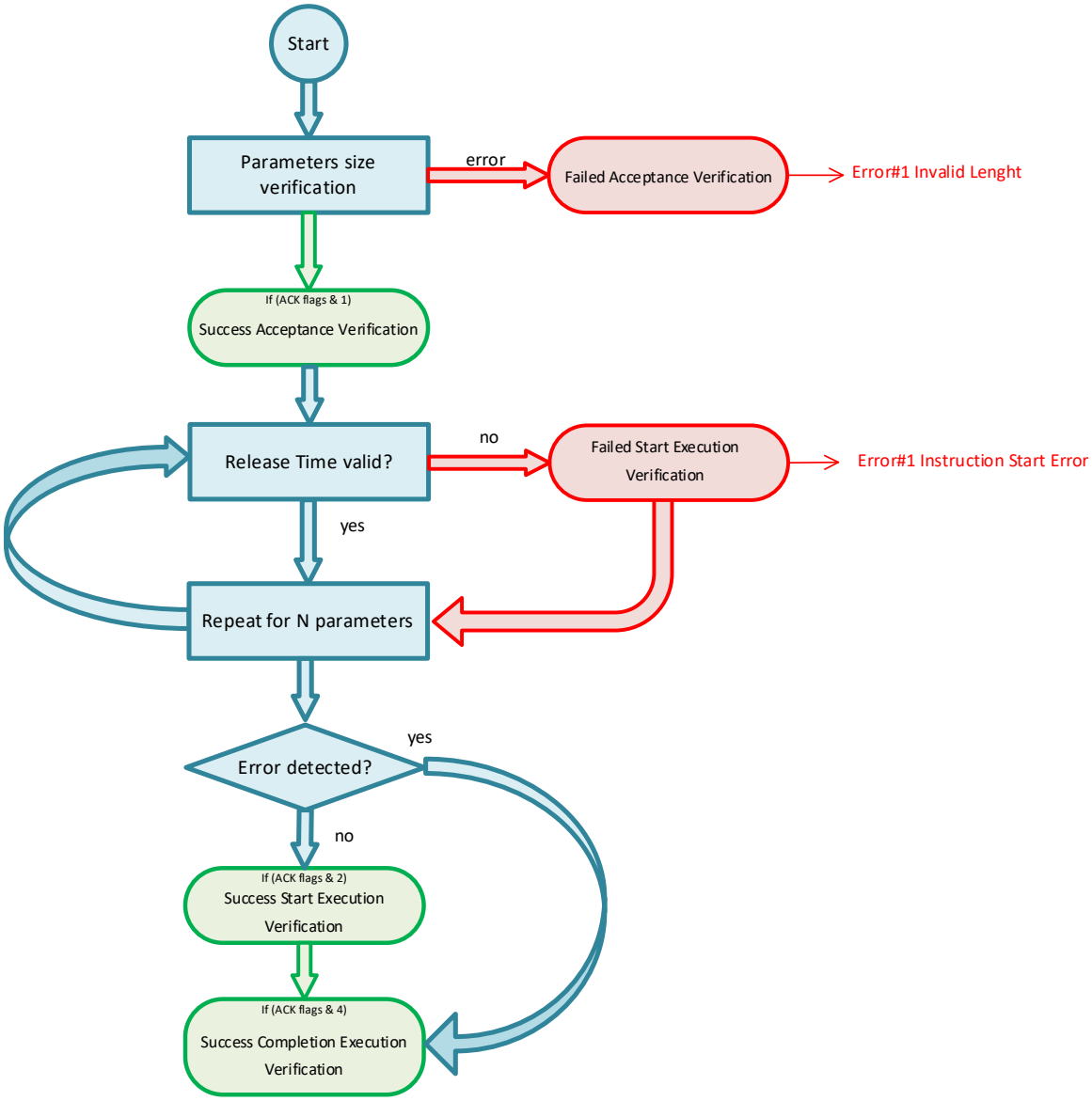
Time-shift scheduled activities identified by request identifier

repeated N times

time offset	N	request ID		
		source ID	application process ID	sequence count
relative time	unsigned integer	enumerated	enumerated	unsigned integer
Mission Absolute Time Format uint32_t (*)	uint16	uint16	uint16	uint16

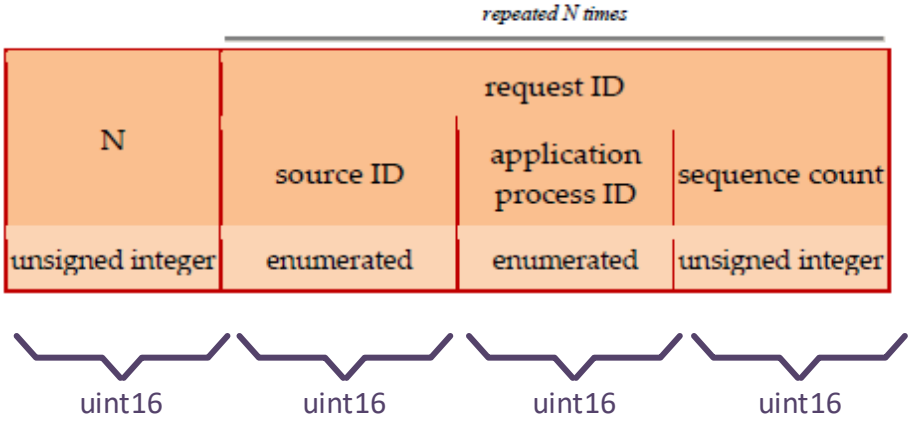
(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

Message request verification flow

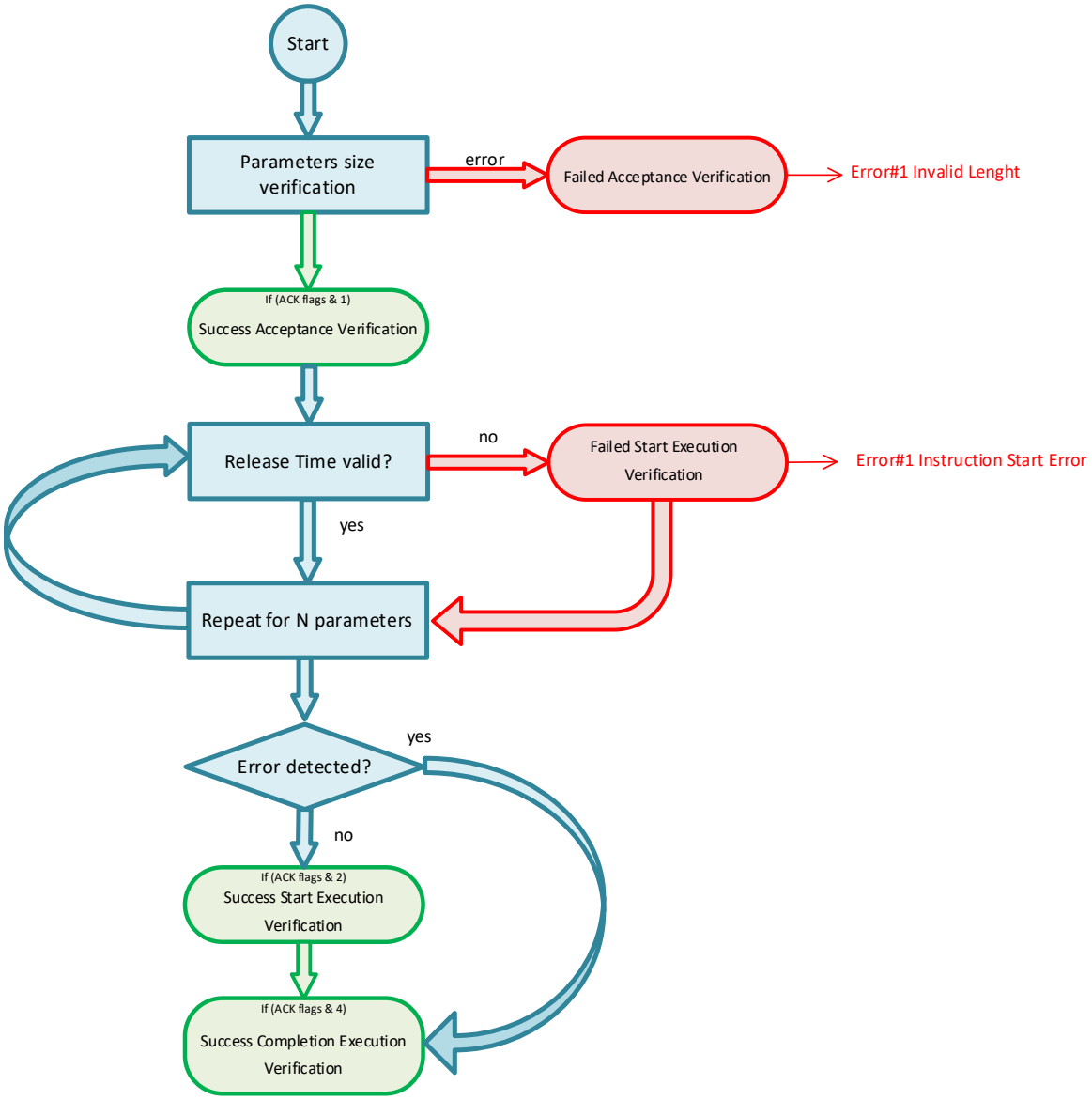


TC[11,9] detail-report time-based scheduled activities identified by request identifier

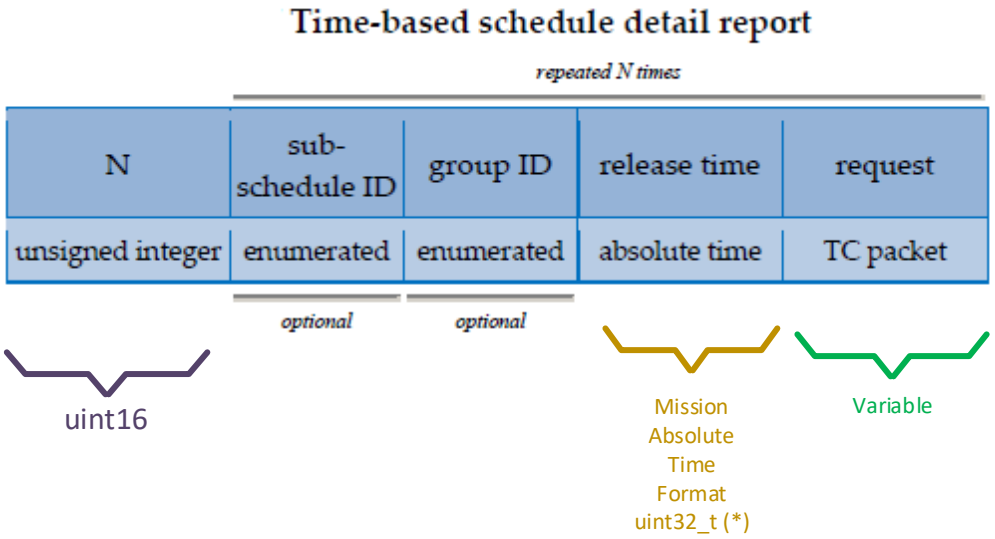
Detail-report time-based scheduled activities identified by request identifier



Message request verification flow

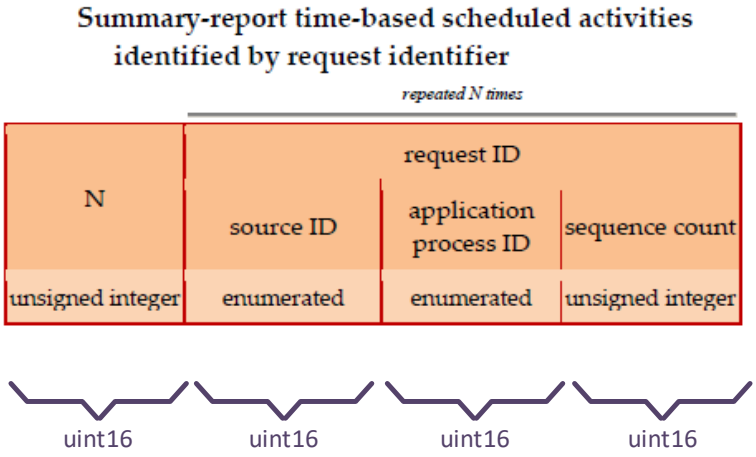


TM[11,10] time-based schedule detail report

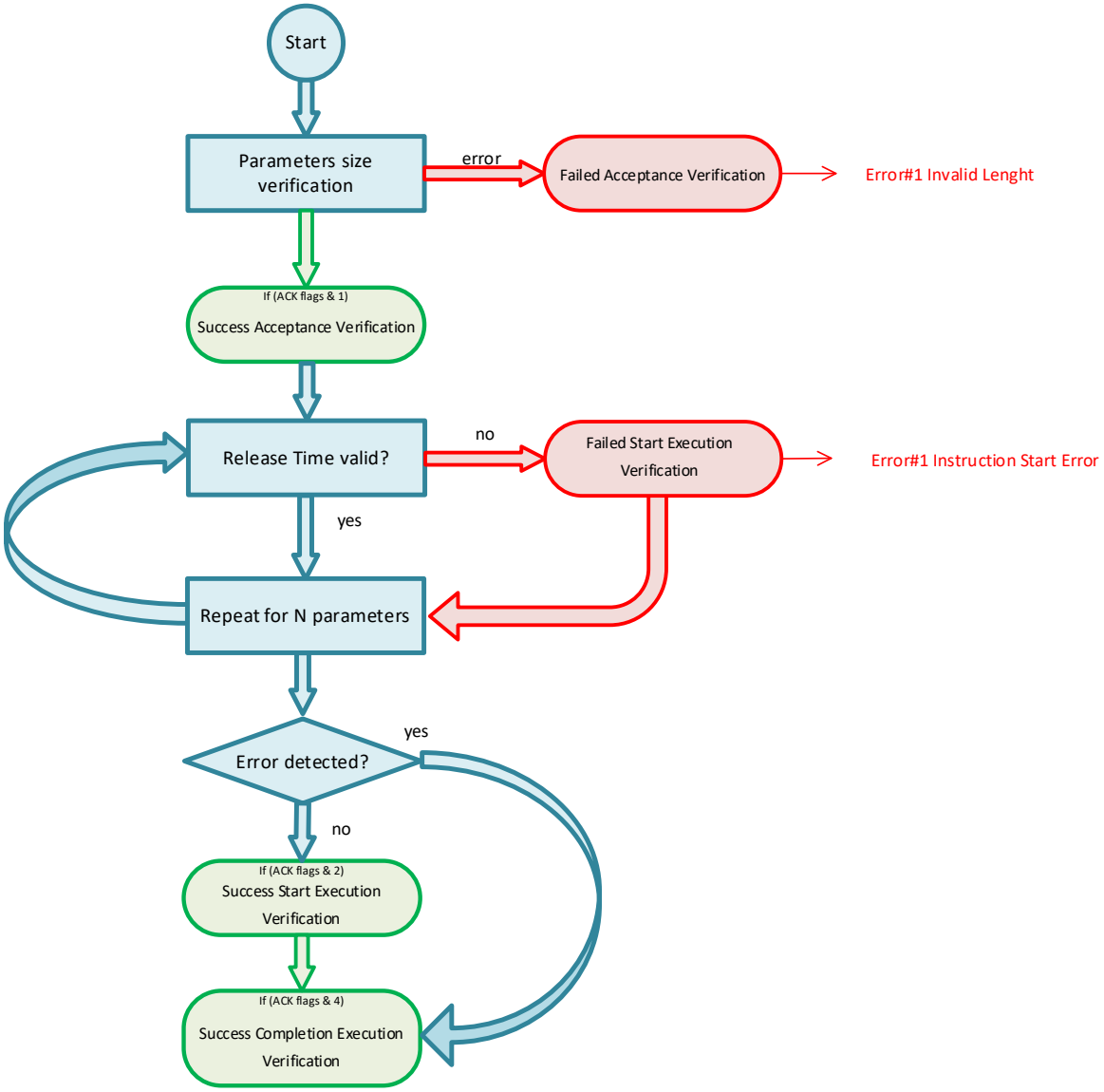


(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

TC[11,12] Summary-report time-based scheduled activities identified by request identifier

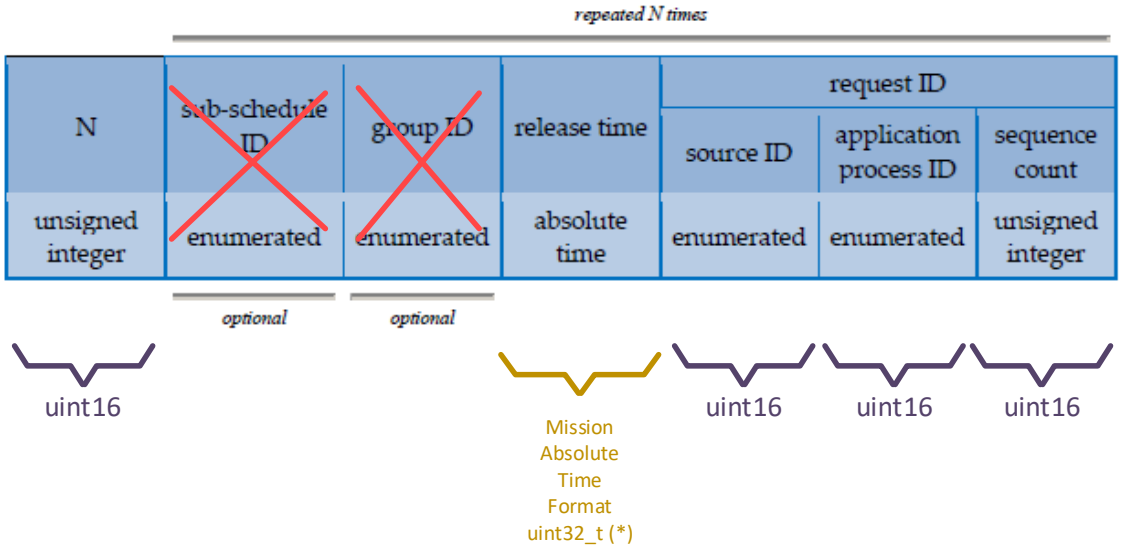


Message request verification flow



TM[11,13] time-based schedule summary report

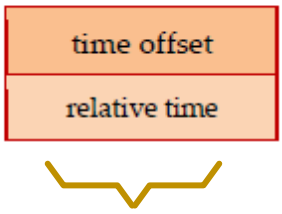
Time-based schedule summary report



(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

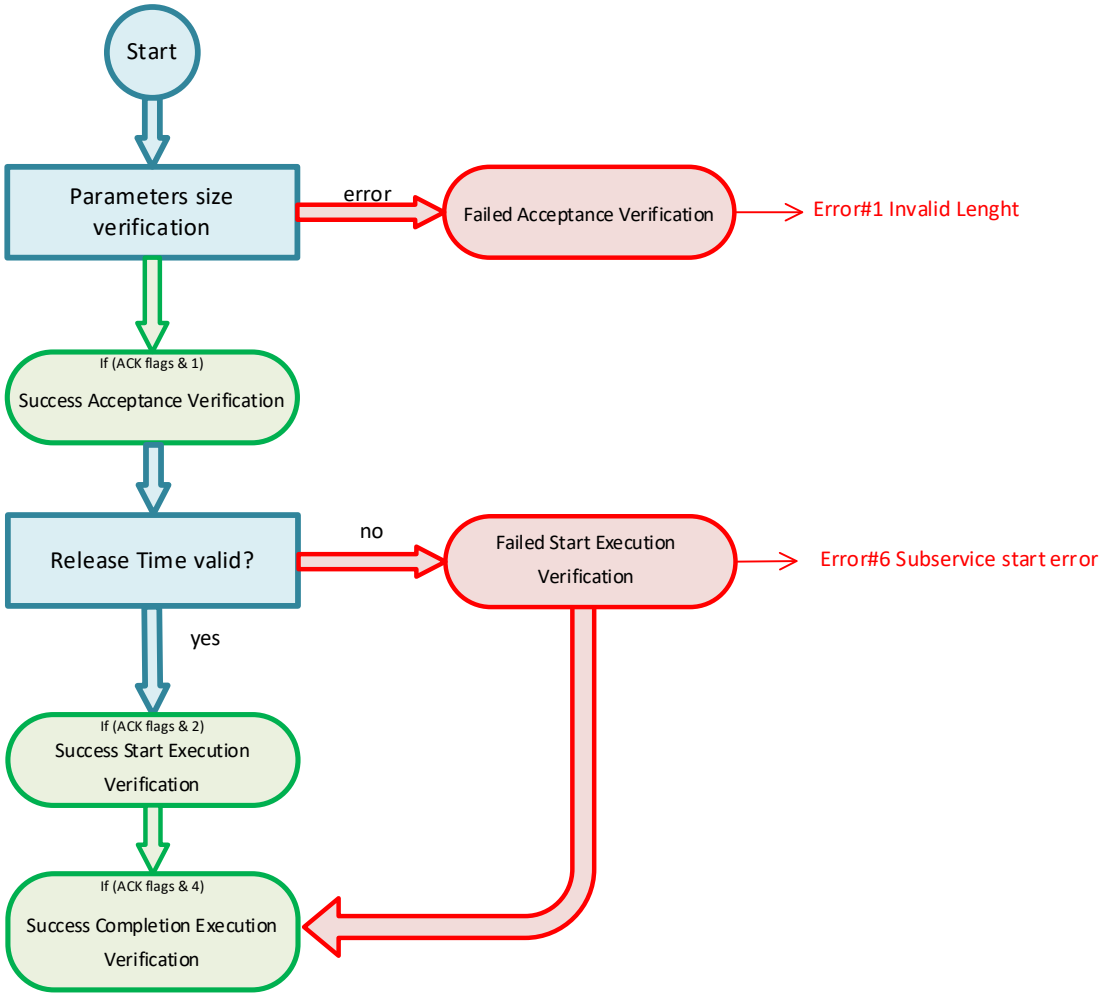
TC[11,15] time-shift all scheduled activities

Time-shift all scheduled activities



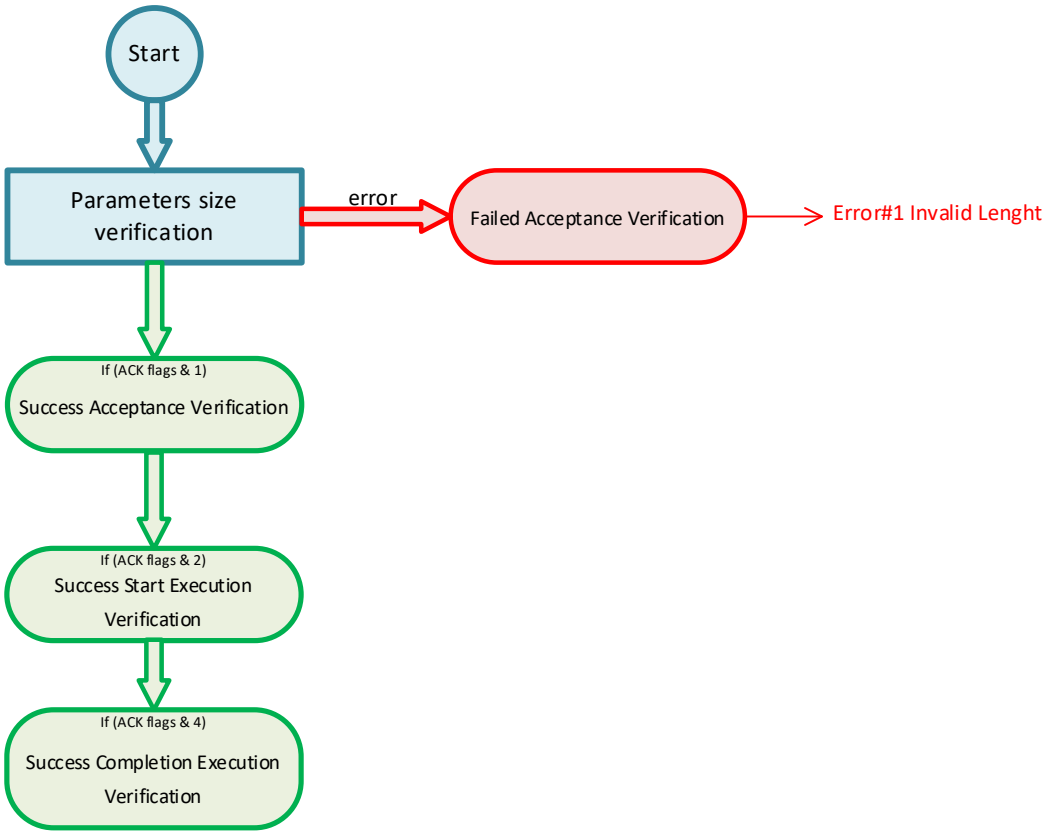
(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

Message request verification flow



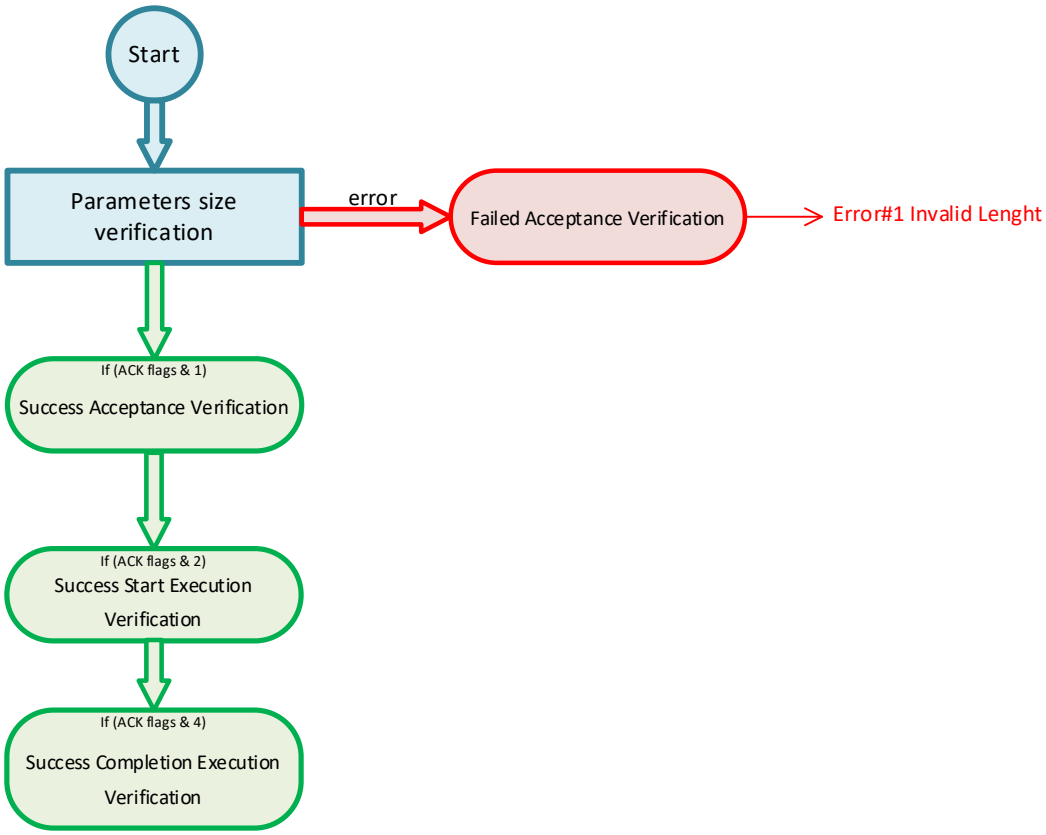
TC[11,16] detail-report all time-based scheduled activities

Message request verification flow



TC[11,17] summary-report all time-based scheduled activities

Message request verification flow



2.13 ST[12] ONBOARD MONITORING SERVICE

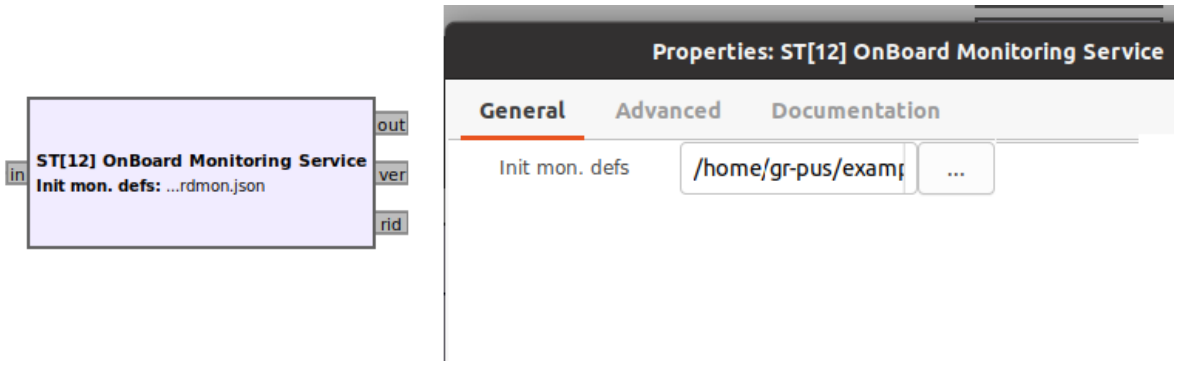
The ***ST[12] OnBoard Monitoring Service*** block will received all message request at its input port and if those request are for service ST[12] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected

The messages with events notification are sent through its rid output port. This messages are composed by the RID number (uint16_t) which trigger the event and a metadata as:
pmt::intern("event") , pmt::from_long(eventType)

The event types are:

Event Code	Event Description

InformativeUnknownEvent = 0	An unknown event occured
LowSeverityUnknownEvent = 4	An unknown anomaly of low severity anomalyhas occurred
MediumSeverityUnknownEvent = 5	An unknown anomaly of medium severity has occurred
HighSeverityUnknownEvent = 6	An unknown anomaly of high severity has occurred



Parameters

(*R*): *Run-time adjustable*

Init monitoring definitions

Path to the json file with the start up monitoring definitions

Messages

In

The message requests input

Out

The message report output

ver

The message verification output, if an error is detected, an output message for the Request Verification Service ST[01] will be addressed

Rid

The message event output

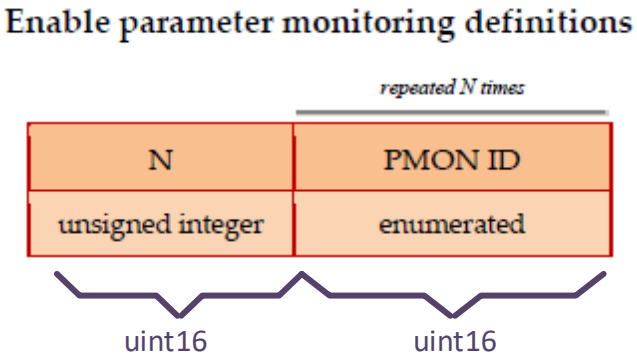
The json init file has next format

```
{
    "enabled" : true,
    "maxTransRepDelay" : 300,
    "monitor": [
        {
            "pmonId": 2,
            "paramId": 1,
            "monInterval" : 10,
            "repetitionNumber": 10,
            "monitoringEnabled": false,
            "definition": [
                {
                    "type": "ValueCheck",
                    "maskValue": 255,
                    "expectedValue": 10,
                    "unexpectedRID": 1
                }
            ]
        },
        {
            "pmonId": 3,
            "paramId": 1,
            "monInterval" : 20,
            "repetitionNumber": 10,
            "monitoringEnabled": false,
            "definition": [
                {
                    "type": "LimitCheck",
                    "lowLimit": 10,
                    "belowLowLimitRID": 1,
                    "highLimit": 15,
                    "aboveHighLimitRID": 2
                }
            ]
        },
```

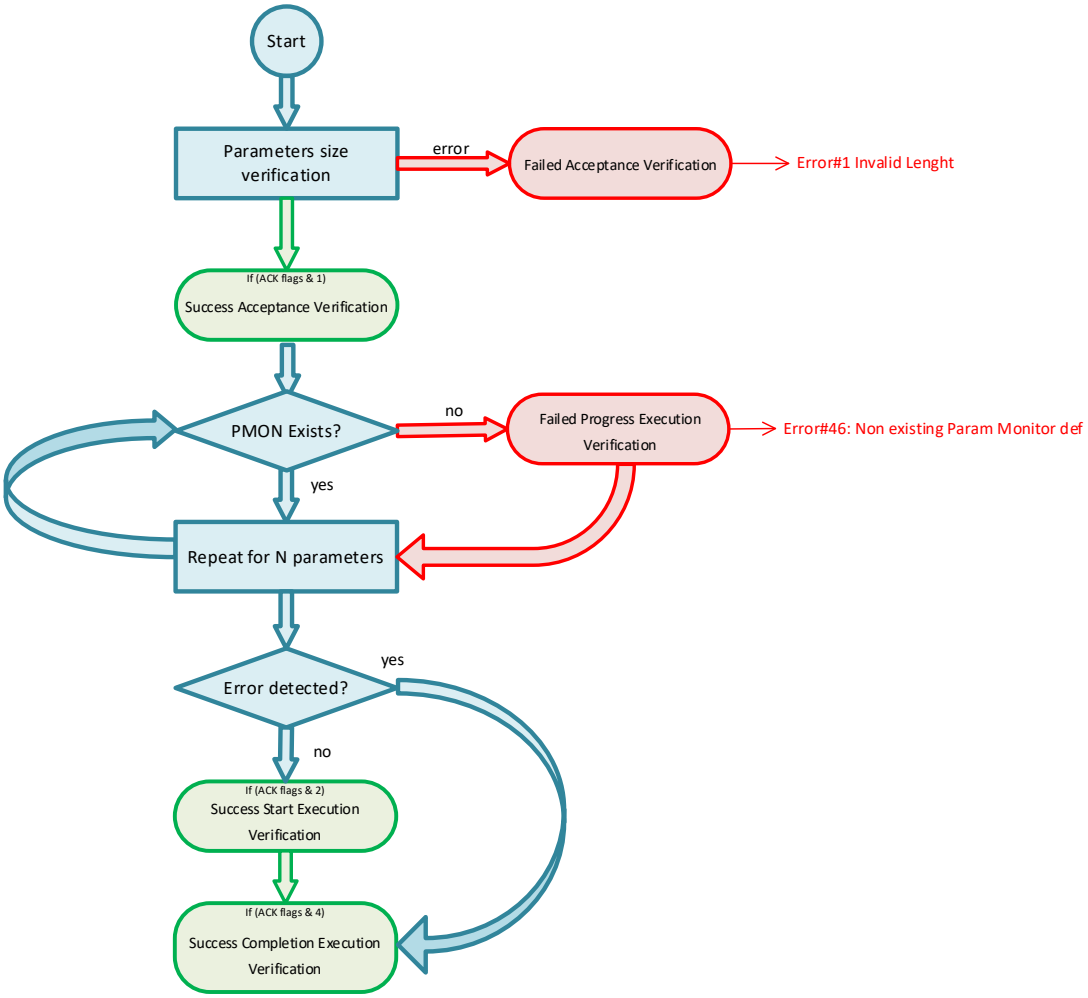
The interval field is expresed in timer tick values, then if timer tick is each 100ms, then “interval” = 10 is 1 sec.

Subtypes requests

TC[12,1] enable parameter monitoring definitions

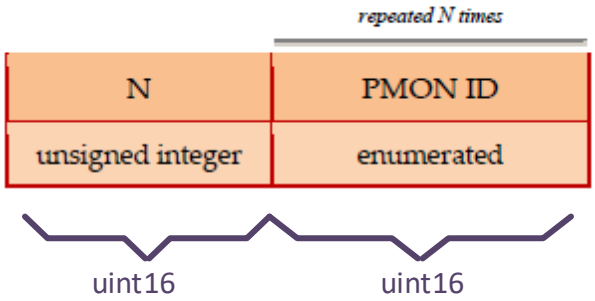


Message request verification flow

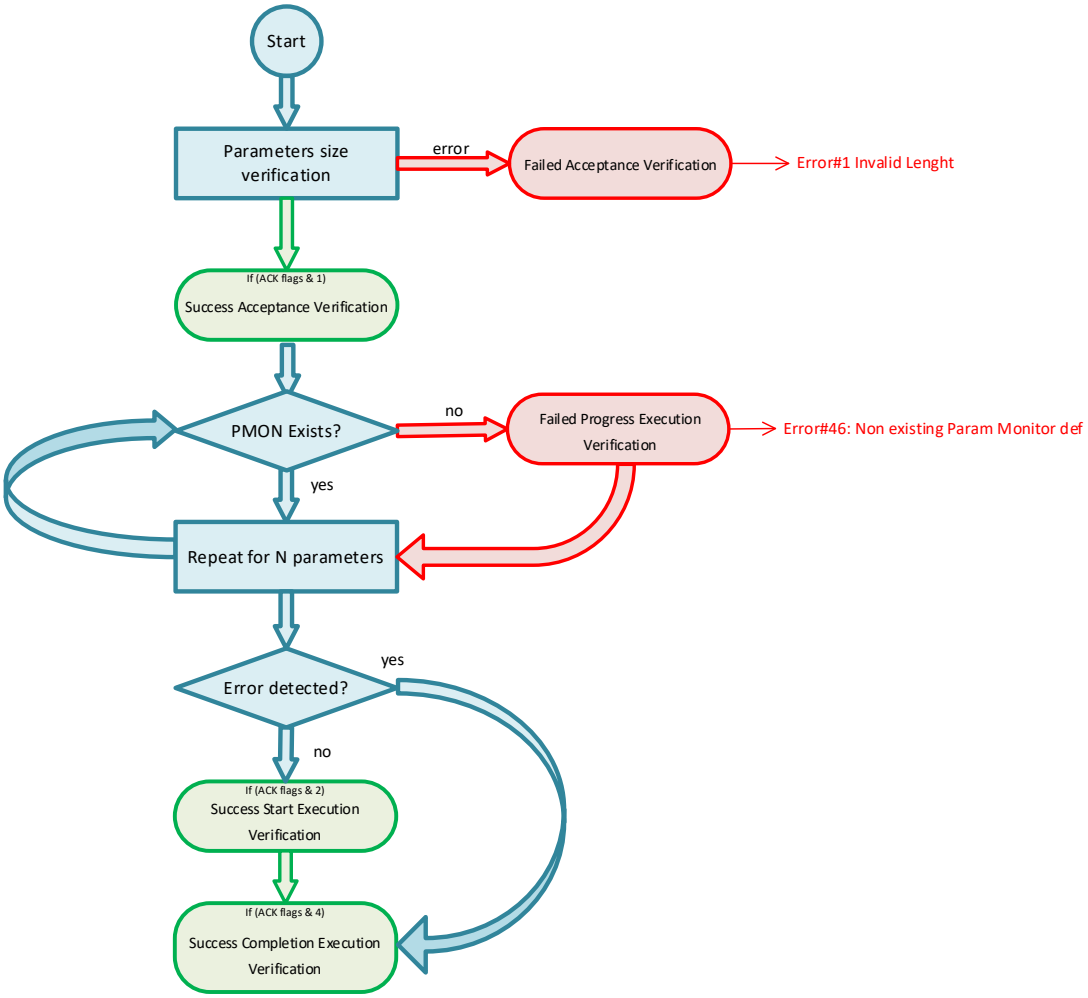


TC[12,2] disable parameter monitoring definitions

Disable parameter monitoring definitions

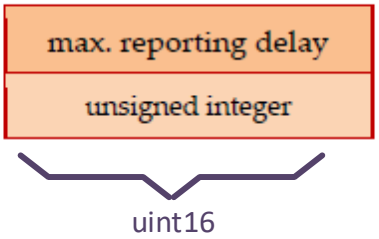


Message request verification flow

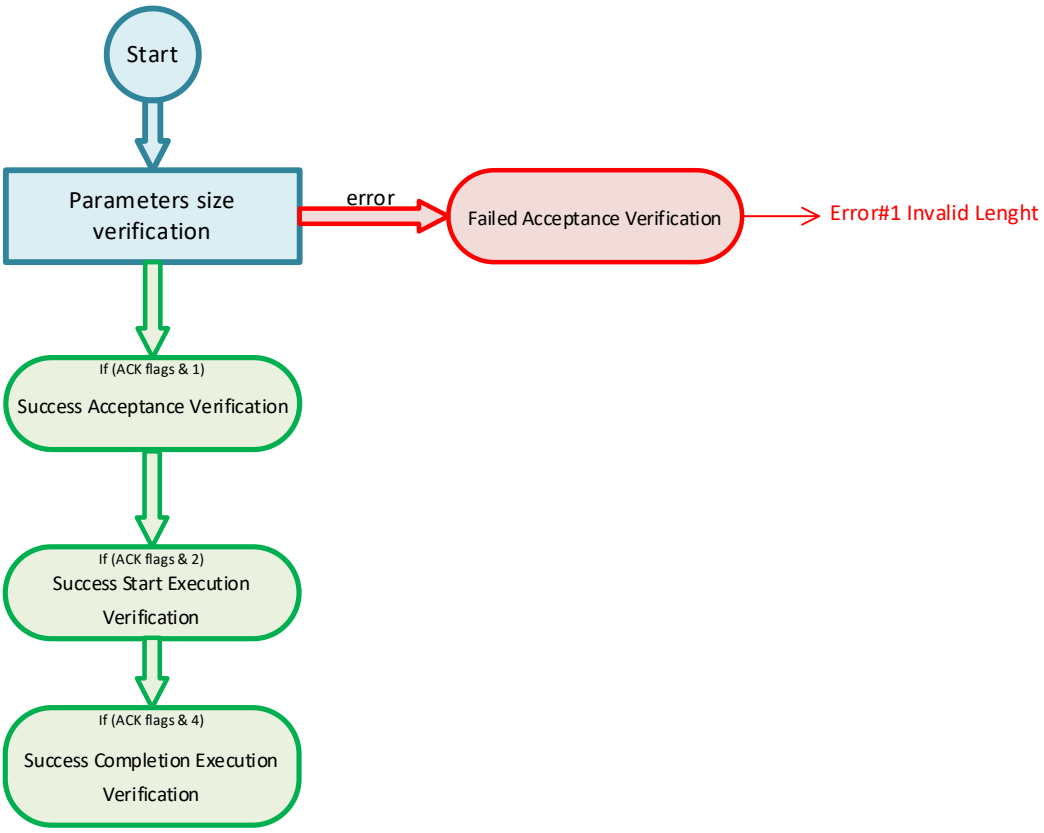


TC[12,3] change the maximum transition reporting delay

Change the maximum transition reporting delay

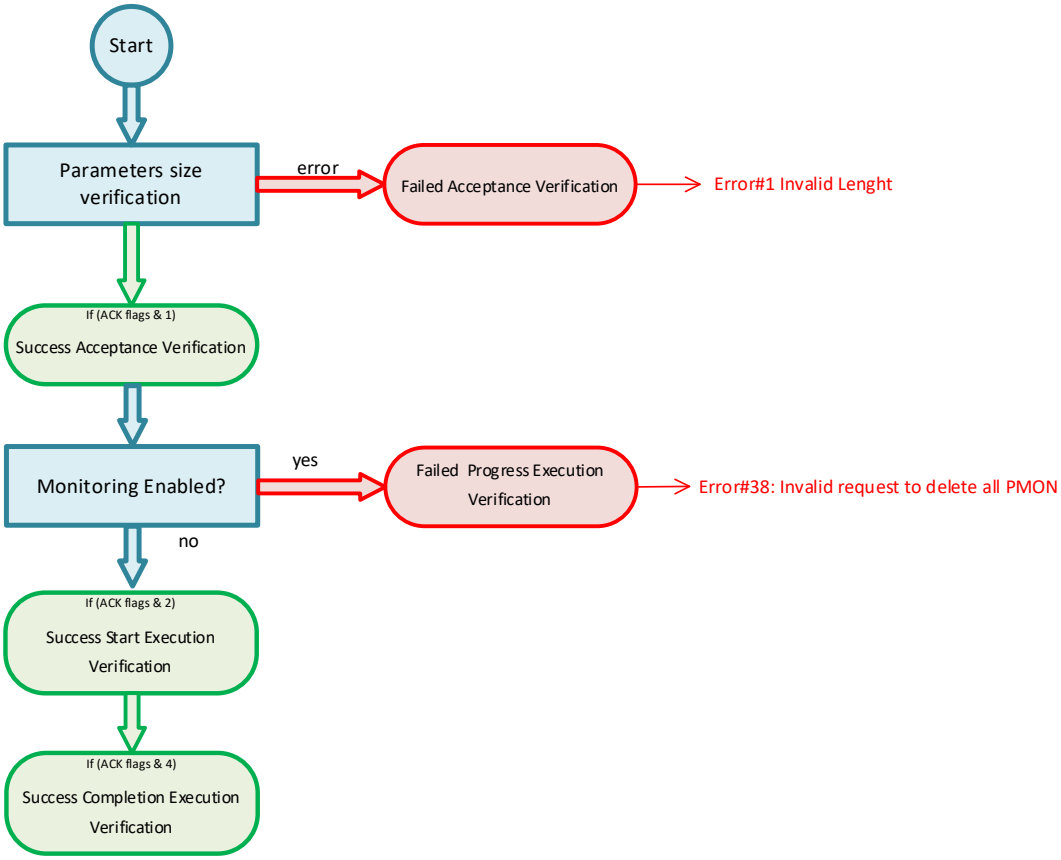


Message request verification flow



TC[12,4] delete all parameter monitoring definitions

Message request verification flow



TC[12,5] add parameter monitoring definitions

Add parameter monitoring definitions

repeated N times

N	PMON ID	monitored parameter ID	check validity condition			monitoring interval	repetition number	check type	check type dependent criteria (see below)
			validity parameter ID	mask	expected value				
unsigned integer	enumerated	enumerated	enumerated	bit-string (deduced size)	deduced	unsigned integer	unsigned integer	enumerated	
			optional			*optional*			
uint16	uint16	uint16	N/A	N/A	N/A	uint16	uint16	uint8	

Add parameter monitoring definitions: expected-value-checking definition fields

mask	spare	expected value	event definition ID
bit-string (deduced size)	bit-string (of event definition ID field size)	deduced	enumerated
	optional		

Param specific — Param specific — uint16

Add parameter monitoring definitions: limit-checking definition fields

low limit	event definition ID	high limit	event definition ID
deduced	enumerated	deduced	enumerated

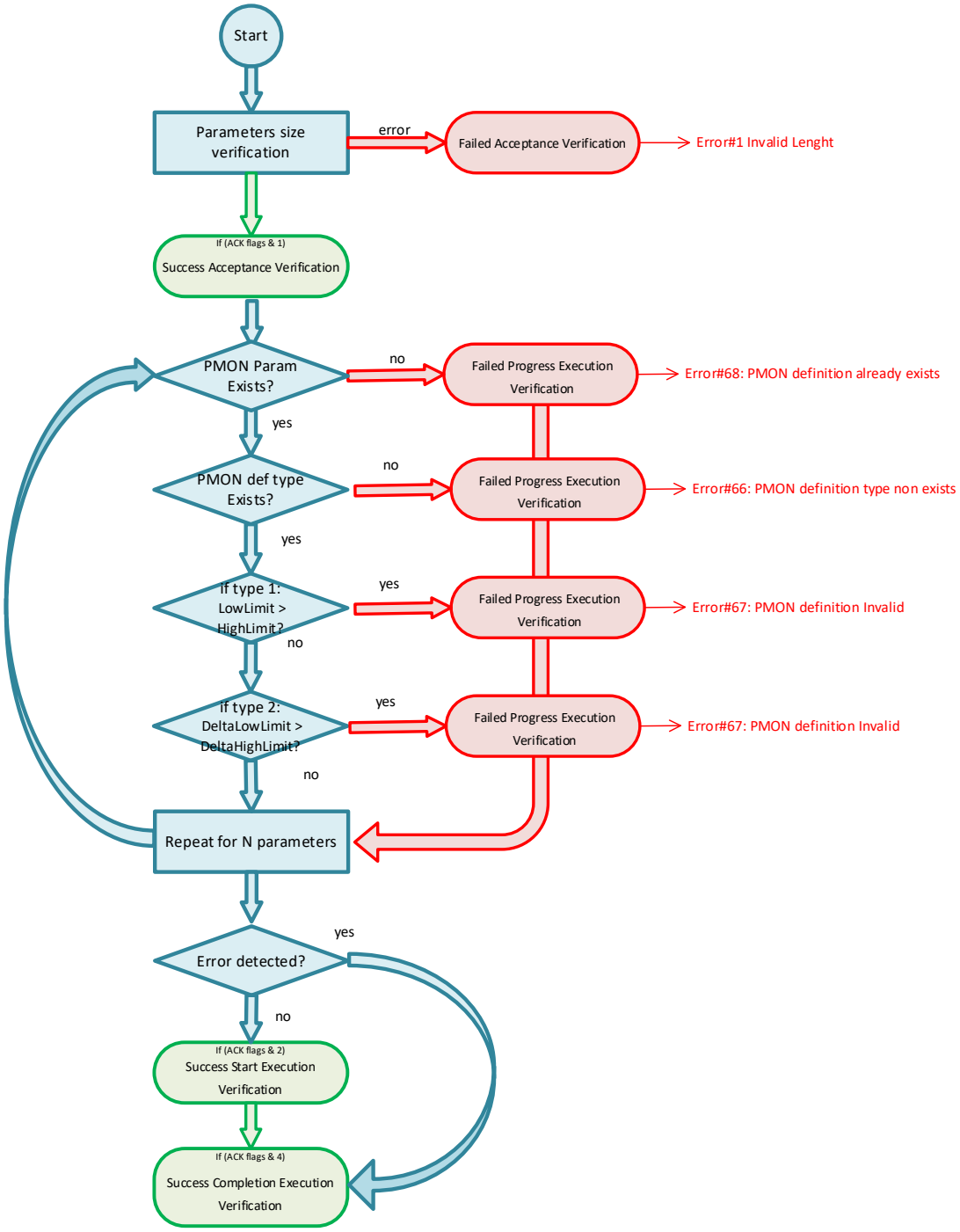
Param specific — uint16 — Param specific — uint16

Add parameter monitoring definitions: delta-checking definition fields

low delta threshold	event definition ID	high delta threshold	event definition ID	number of consecutive delta values
deduced	enumerated	deduced	enumerated	unsigned integer

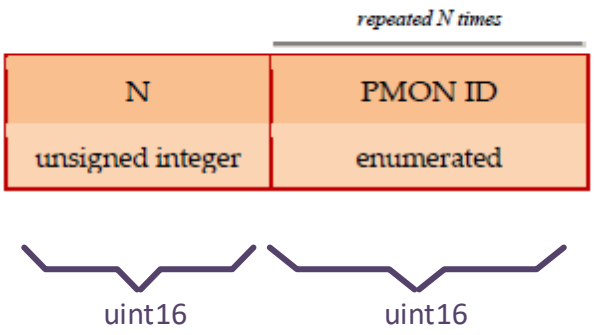
Param specific — uint16 — Param specific — uint16 — uint16

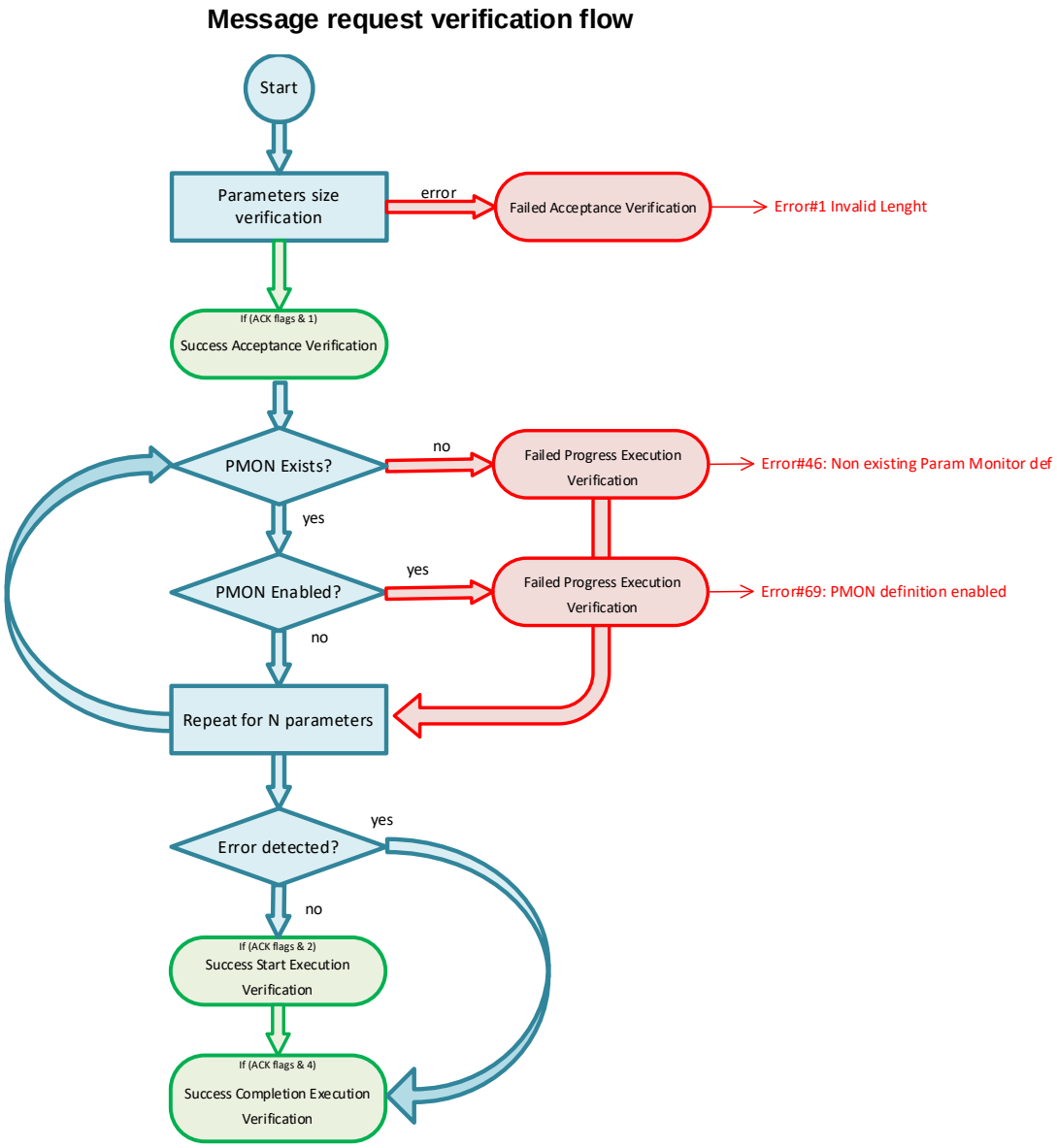
Message request verification flow



TC[12,6] delete parameter monitoring definitions

Delete parameter monitoring definitions





TC[12,7] modify parameter monitoring definitions

Modify parameter monitoring definitions

repeated N times

N	PMON ID	monitored parameter ID	repetition number	check type	check type dependent criteria (see below)
unsigned integer	enumerated	enumerated	unsigned integer	enumerated	

uint16 uint16 uint16 uint16 uint8

Modify parameter monitoring definitions: expected-value-checking definition fields

mask	spare	expected value	event definition ID
bit-string (deduced size)	bit-string (of event definition ID field size)	deduced	enumerated

optional

Param specific — Param specific — uint16

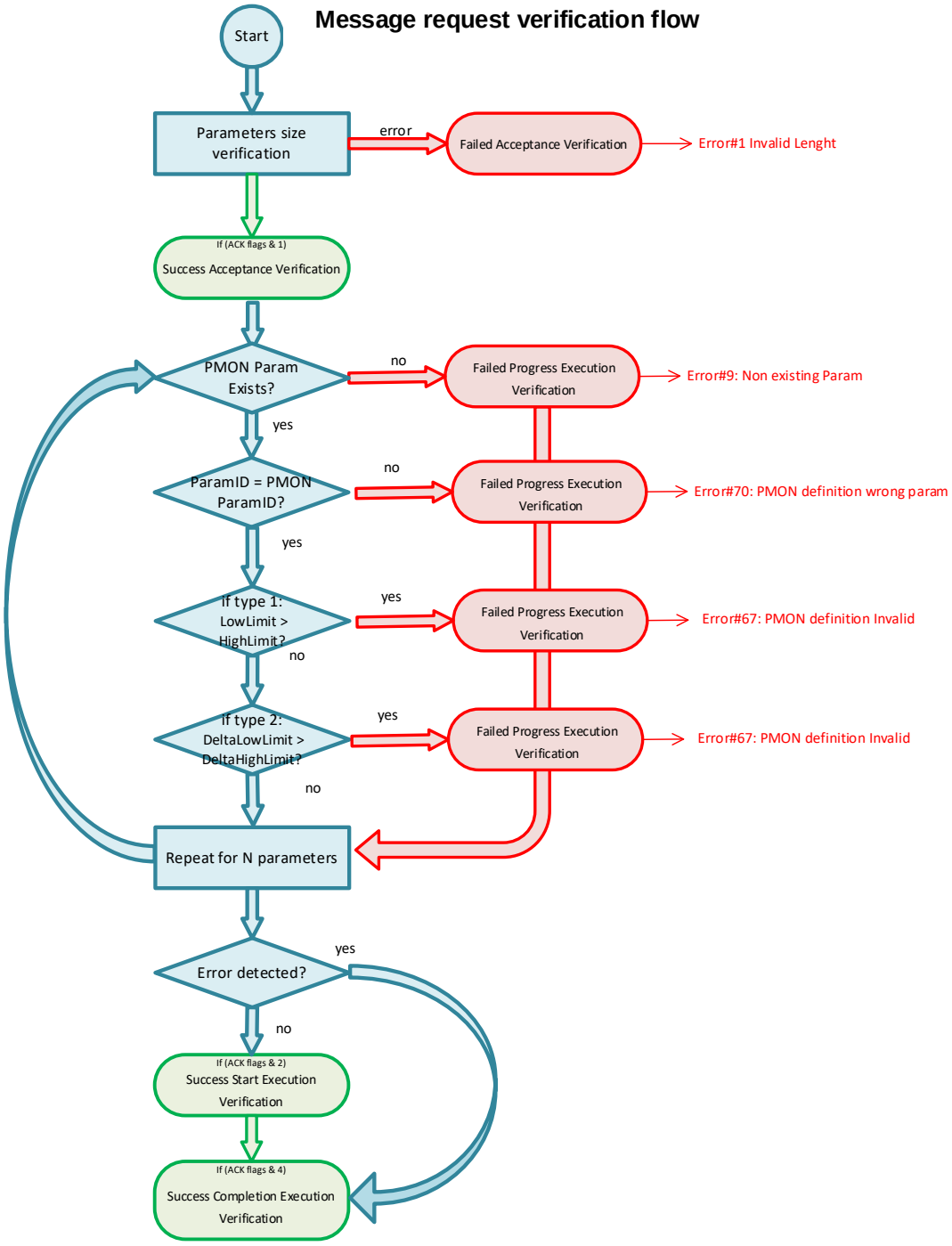
Modify parameter monitoring definitions: limit-checking definition fields

low limit	event definition ID	high limit	event definition ID
deduced	enumerated	deduced	enumerated

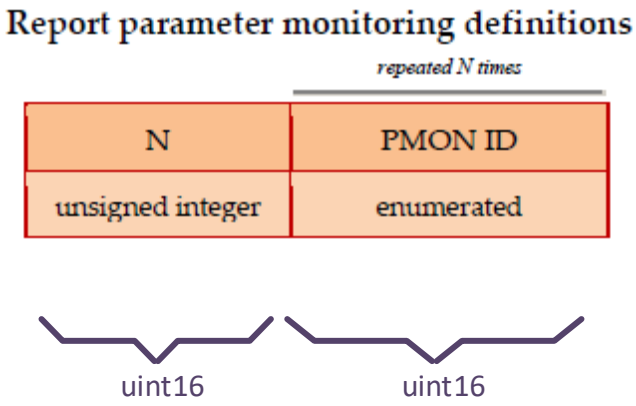
Param specific — uint16 — Param specific — uint16

Modify parameter monitoring definitions: limit-checking definition fields

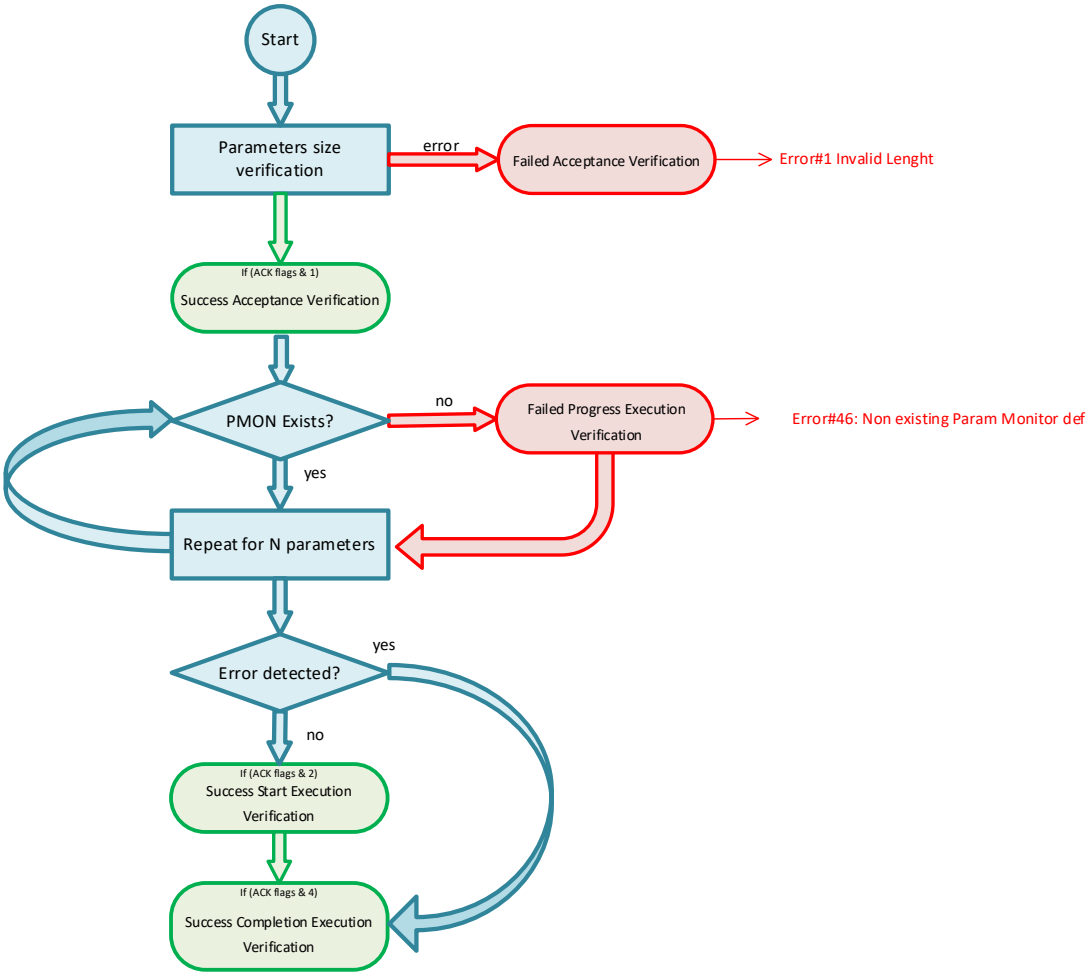
low delta threshold	event definition ID	high delta threshold	event definition ID	number of consecutive delta values
deduced	enumerated	deduced	enumerated	unsigned integer



TC[12,8] report parameter monitoring definitions

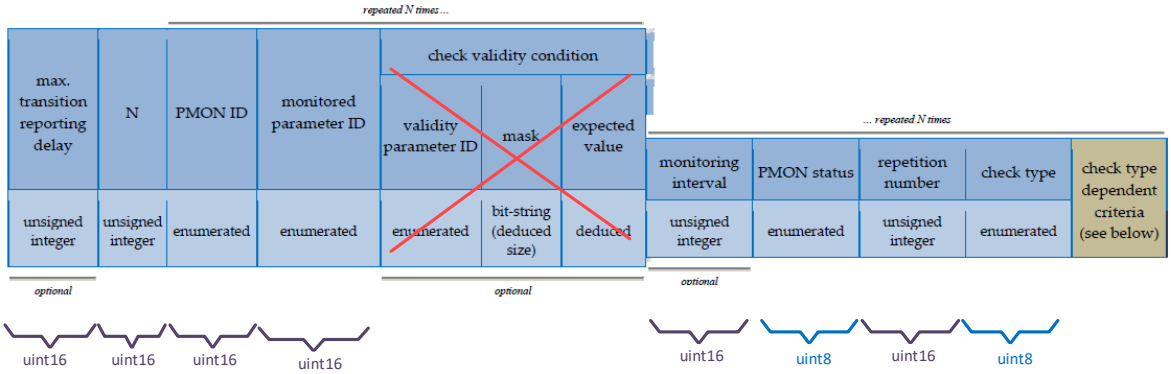


Message request verification flow



TM[12,9] parameter monitoring definition report

Parameter monitoring definition report



Parameter monitoring definition report: expected-value-checking definition fields

mask	spare	expected value	event definition ID
bit-string (deduced size)	bit-string (of event definition ID field size)	deduced	enumerated

optional

Param specific

Param specific

uint16

Parameter monitoring definition report: limit-checking definition fields

low limit	event definition ID	high limit	event definition ID
deduced	enumerated	deduced	enumerated

Param specific

uint16

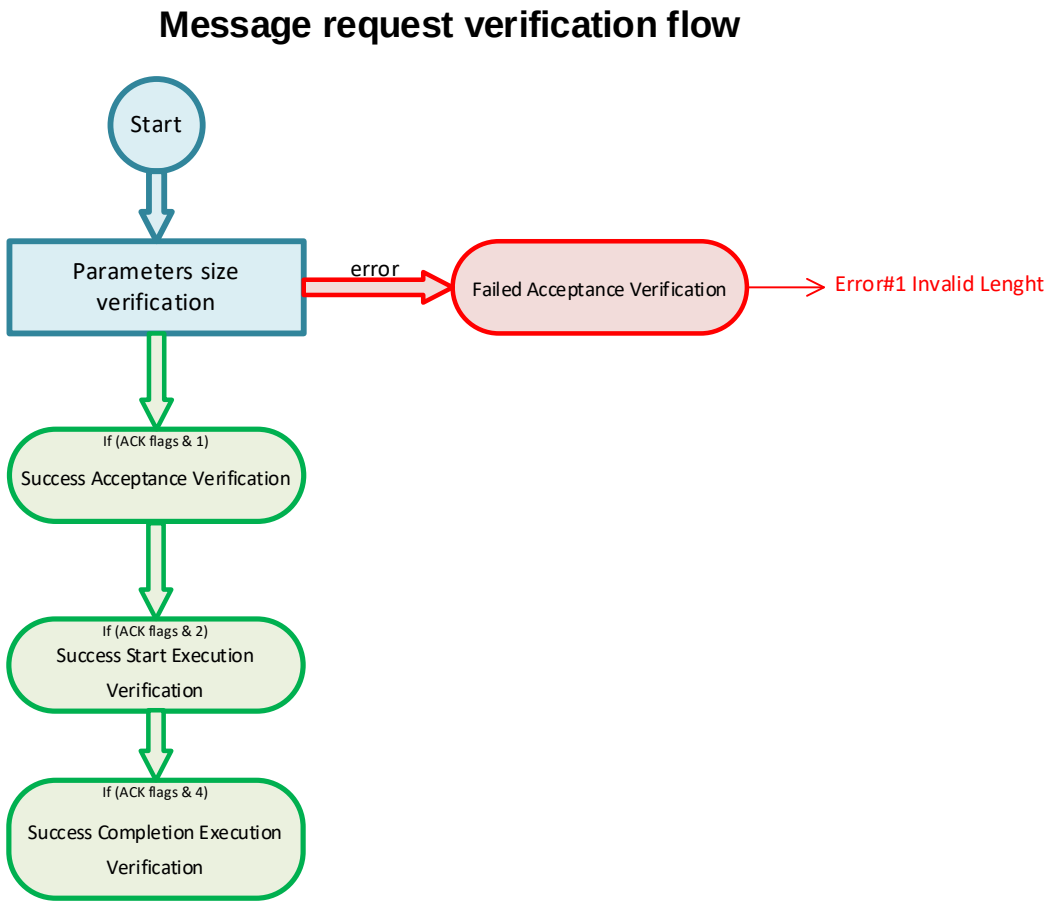
Param specific

uint16

Selected parameter monitoring definition list: delta-checking definition fields

low delta threshold	event definition ID	high delta threshold	event definition ID	number of consecutive delta values
deduced	enumerated	deduced	enumerated	unsigned integer

TC[12,10] report the out-of-limits



TM[12,11] out-of-limits report

Out-of-limits report

repeated N times

N	PMON ID	monitored parameter ID	check type	expected value check mask	parameter value	limit crossed	previous PMON checking status	current PMON checking status	transition time
unsigned integer	enumerated	enumerated	enumerated	bit-string (deduced size)	deduced	deduced	enumerated	enumerated	absolute time

deduced presencce

uint16 uint16 uint16 uint8 Param specific Param specific Param specific uint8 uint8 Mission Absolute Time Format uint32_t (*)

(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

TM[12,12] check transition report

Check transition report

repeated N times

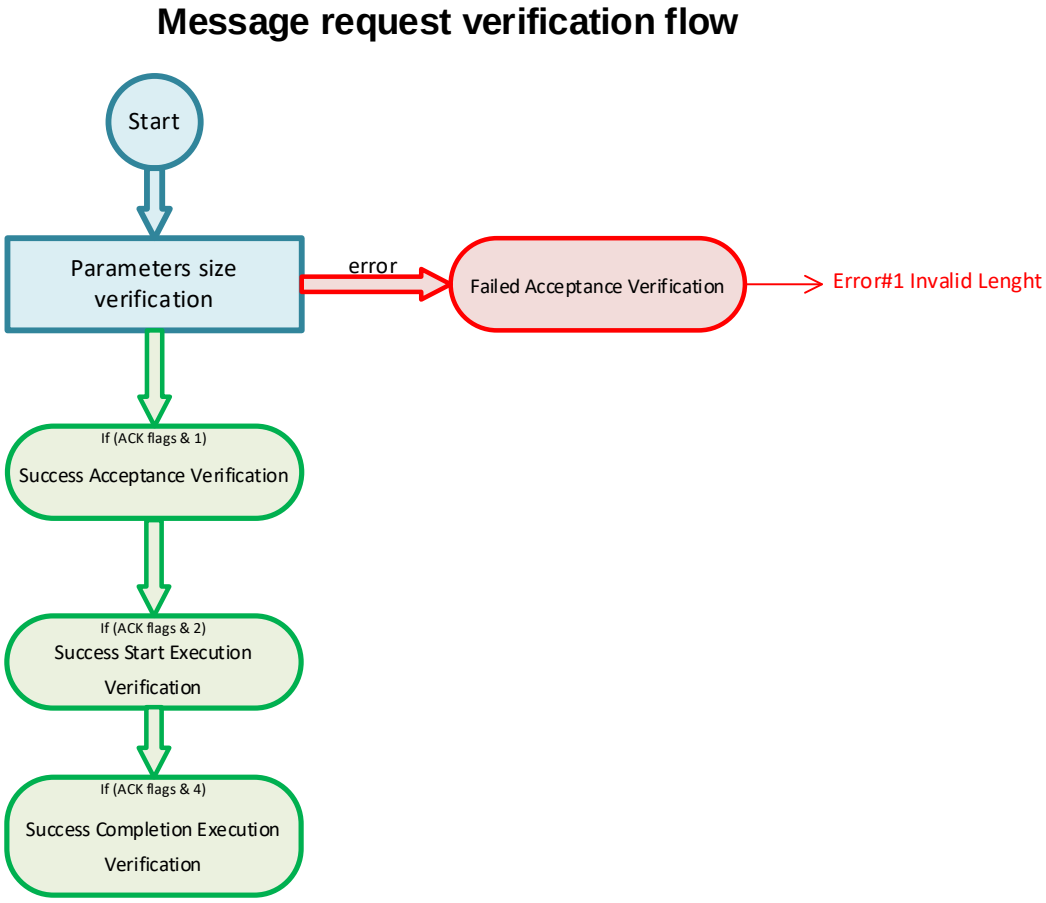
N	PMON ID	monitored parameter ID	check type	expected value check mask	parameter value	limit crossed	previous PMON checking status	current PMON checking status	transition time
unsigned integer	enumerated	enumerated	enumerated	bit-string (deduced size)	deduced	deduced	enumerated	enumerated	absolute time

deduced presence

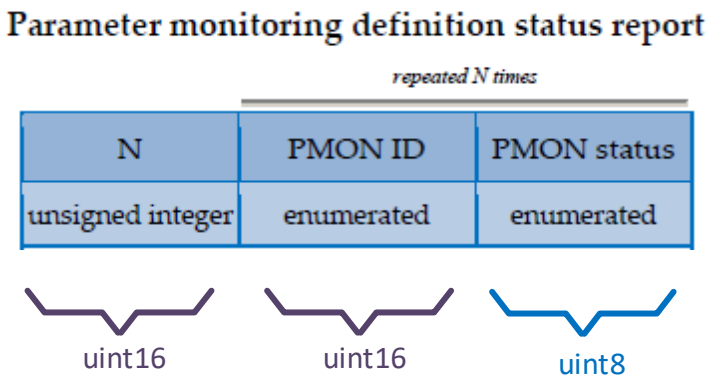
uint16 uint16 uint16 uint8 Param specific Param specific Param specific uint8 uint8 Mission Absolute Time Format uint32_t (*)

(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

TC[12,13] report the status of each parameter monitoring definition



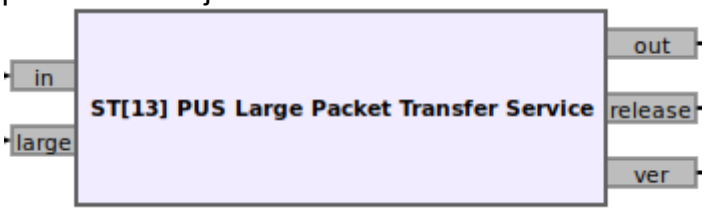
TM[12,14] parameter monitoring definition status report



2.14[ST13]LARGE PACKET TRANSFER

The ***ST[13] Large Packet Transfer Service*** block will received all message request at its input port and if those request are for service ST[13] and for a valid

subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Messages

In

The message requests input

Out

The message report output

large

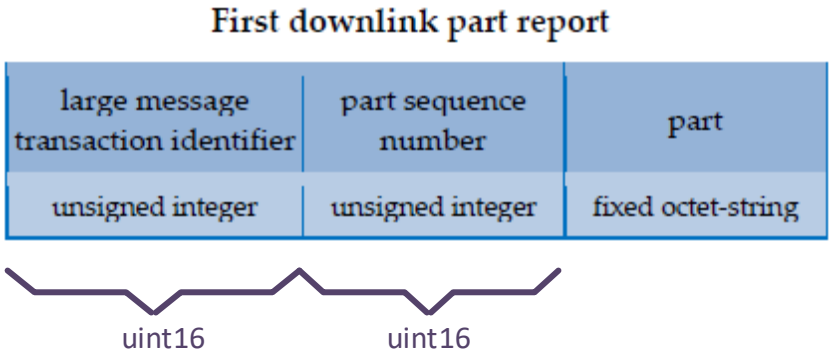
The large message input, any message received at this input will be splitted and outputted thru the Out message port

release

The block will outputed all the messages requests than arrived throught the in port as message request part after join then

Subtypes requests

TM[13,1] first downlink part report (for the first part)



TM[13,2] intermediate downlink part report (for the intermediate parts)

Intermediate downlink part report

large message transaction identifier	part sequence number	part
unsigned integer	unsigned integer	fixed octet-string

uint16 uint16

TM[13,3] last downlink part report (for the last part)

Last downlink part report

large message transaction identifier	part sequence number	part
unsigned integer	unsigned integer	fixed octet-string of deduced size
NOTE The size of the part field is deduced from the size of the telemetry packet that is transported.		

uint16 uint16

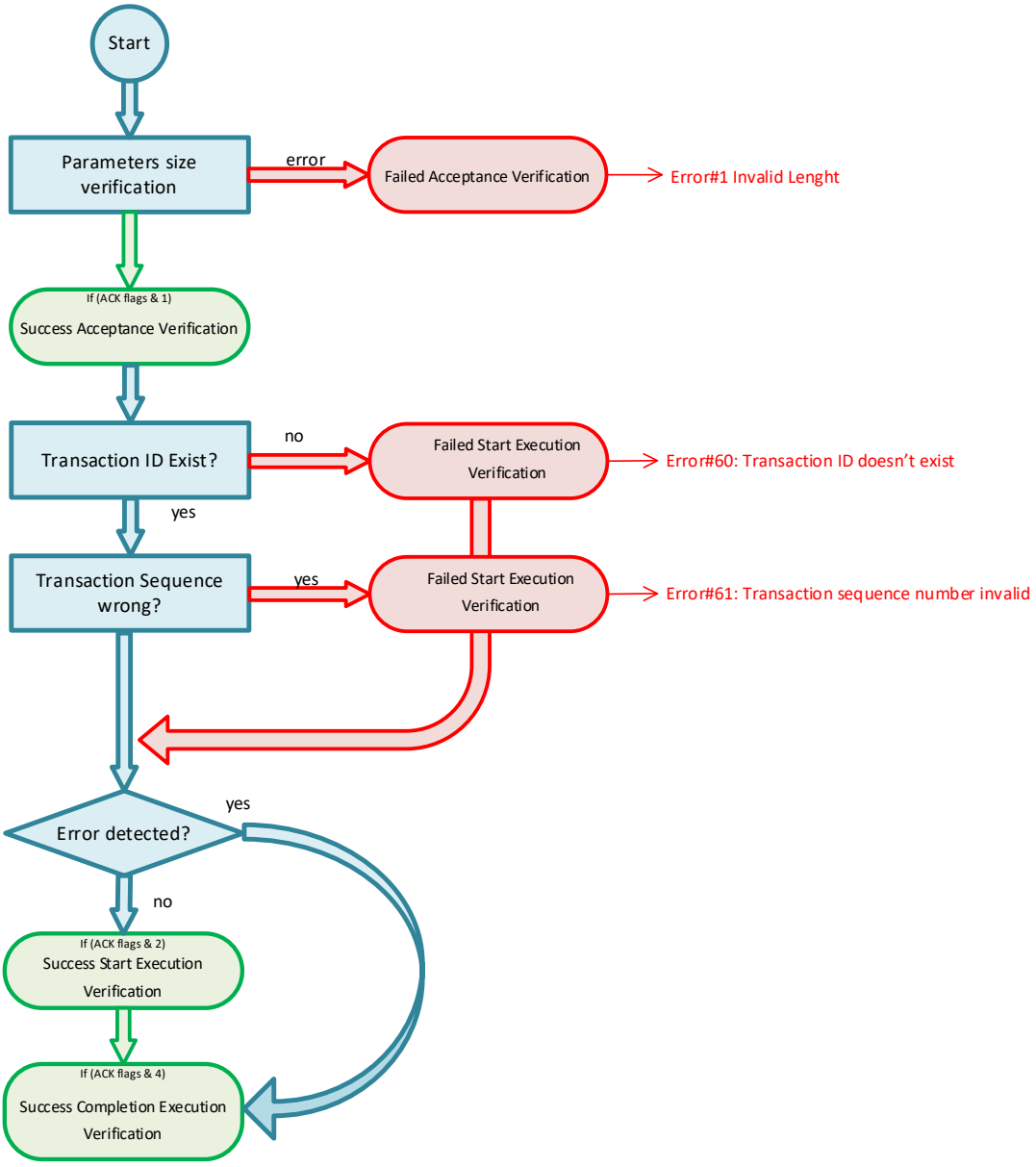
TC[13,9] uplink the first part (for the first part)

Uplink the first part

large message transaction identifier	part sequence number	part
unsigned integer	unsigned integer	fixed octet-string

uint16 uint16

Message request verification flow

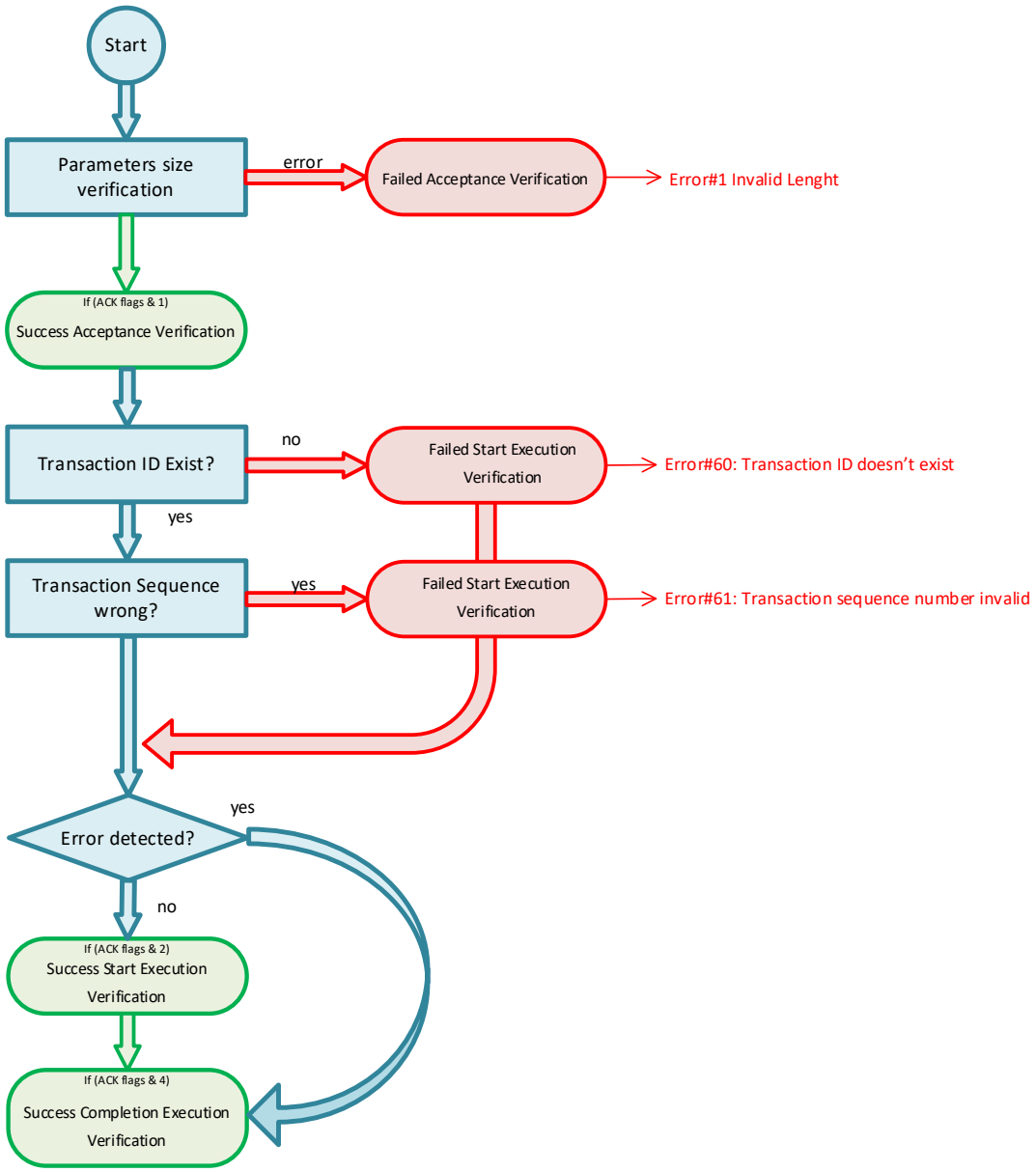


TC[13,10] uplink an intermediate part (for the intermediate parts)

Uplink an intermediate part

large message transaction identifier	part sequence number	part
unsigned integer	unsigned integer	fixed octet-string

Message request verification flow

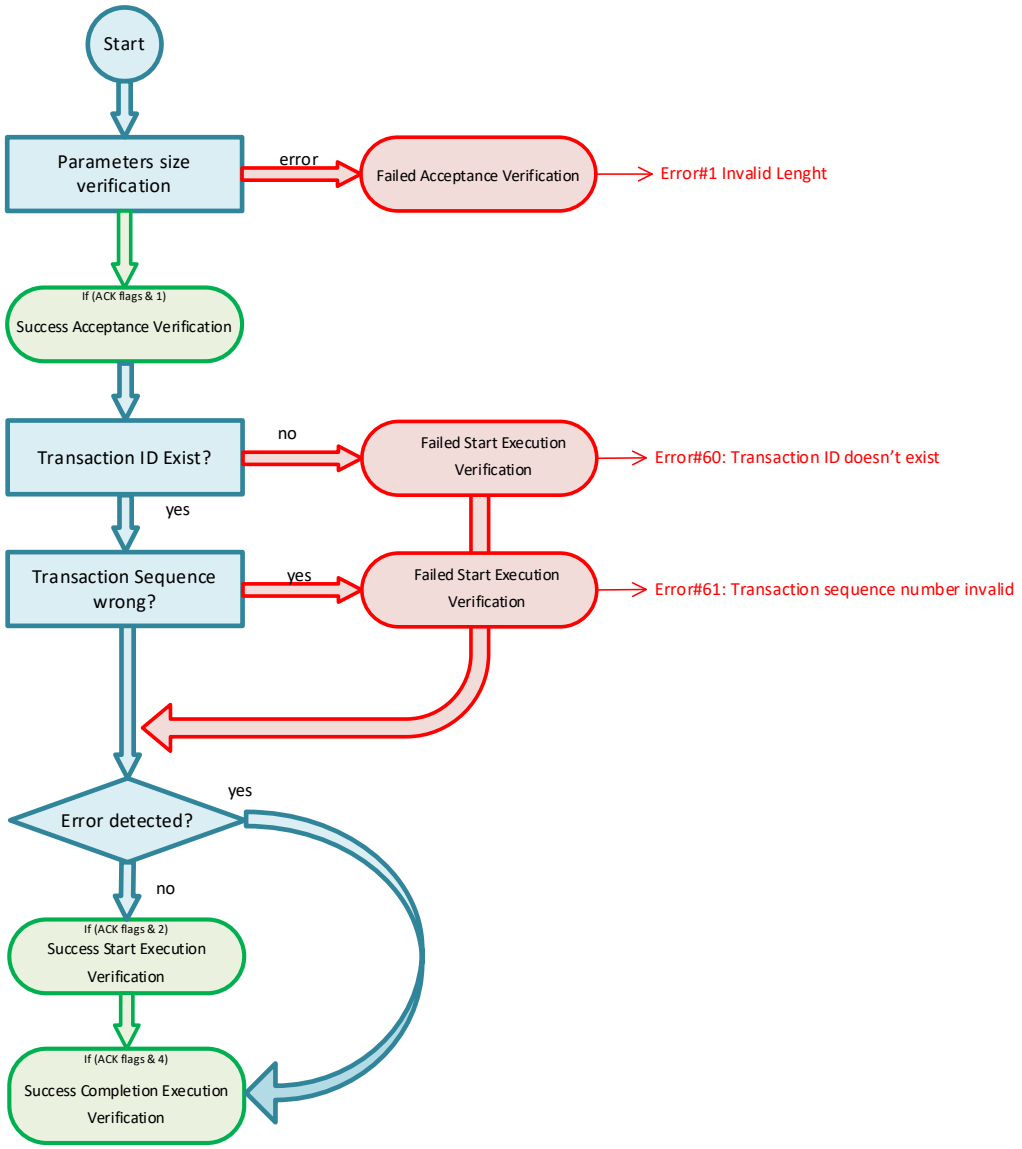


TC[13,11] uplink the last part (for the last part)

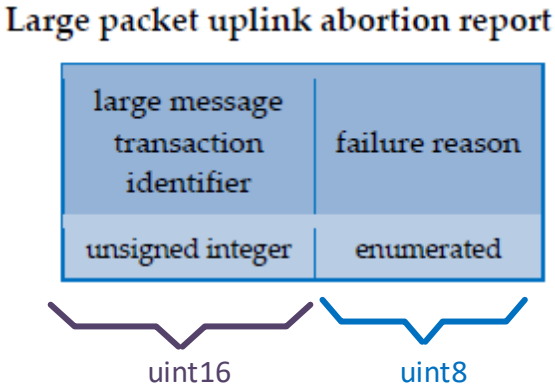
Uplink the last part

large message transaction identifier	part sequence number	part
unsigned integer	unsigned integer	fixed octet-string of deduced size
NOTE The size of the part field is deduced from the size of the large telecommand packet that is transported.		

Message request verification flow



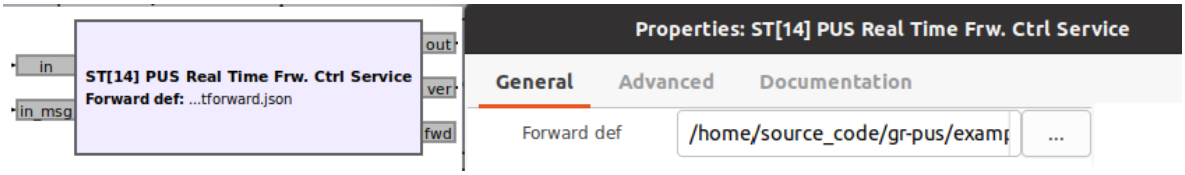
TM[13,16] large packet uplink abortion report



Failure Error Type	Error Code	Error Description
LargePacketUplinkAbortErrorType	LargeUplinkNoPacketFound = 0	In the Large Packet Transfer service, when traying to joint and non existing largeMessageTransactionIdentifier
	LargeUplinkTimeoutExpiress = 1	In the Large Packet Transfer service, when the timeout between uplink part expires
	LargeUplinkMissingPart = 2	In the Large Packet Transfer service, when traying to joint and packet with missing parts

2.15 [ST14]REAL TIME FORWARDING CONTROL

The ***ST[14] Real Time Forwarding Control Service*** block will received all message request at its input port and if those request are for service ST[14] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Forward defi

Path to the json file with the start up forward definitions, left empty if no init required

Messages

In

The message requests input

Out

The message report output

in_msg

fwd

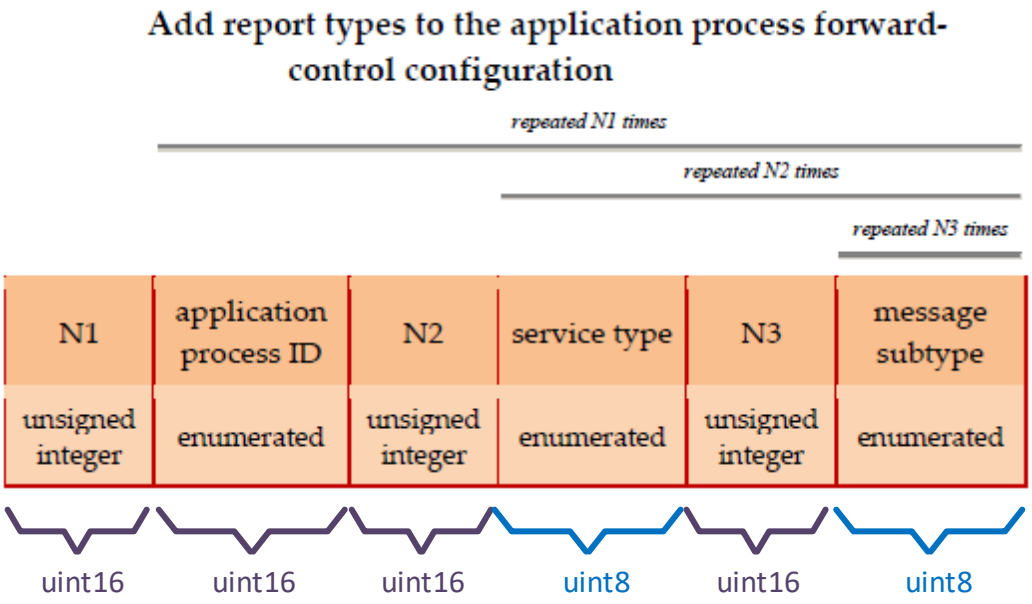
The in_msg Input will receive the message reports, any reports matching the forwarding definitions will be sent thru the fwd port

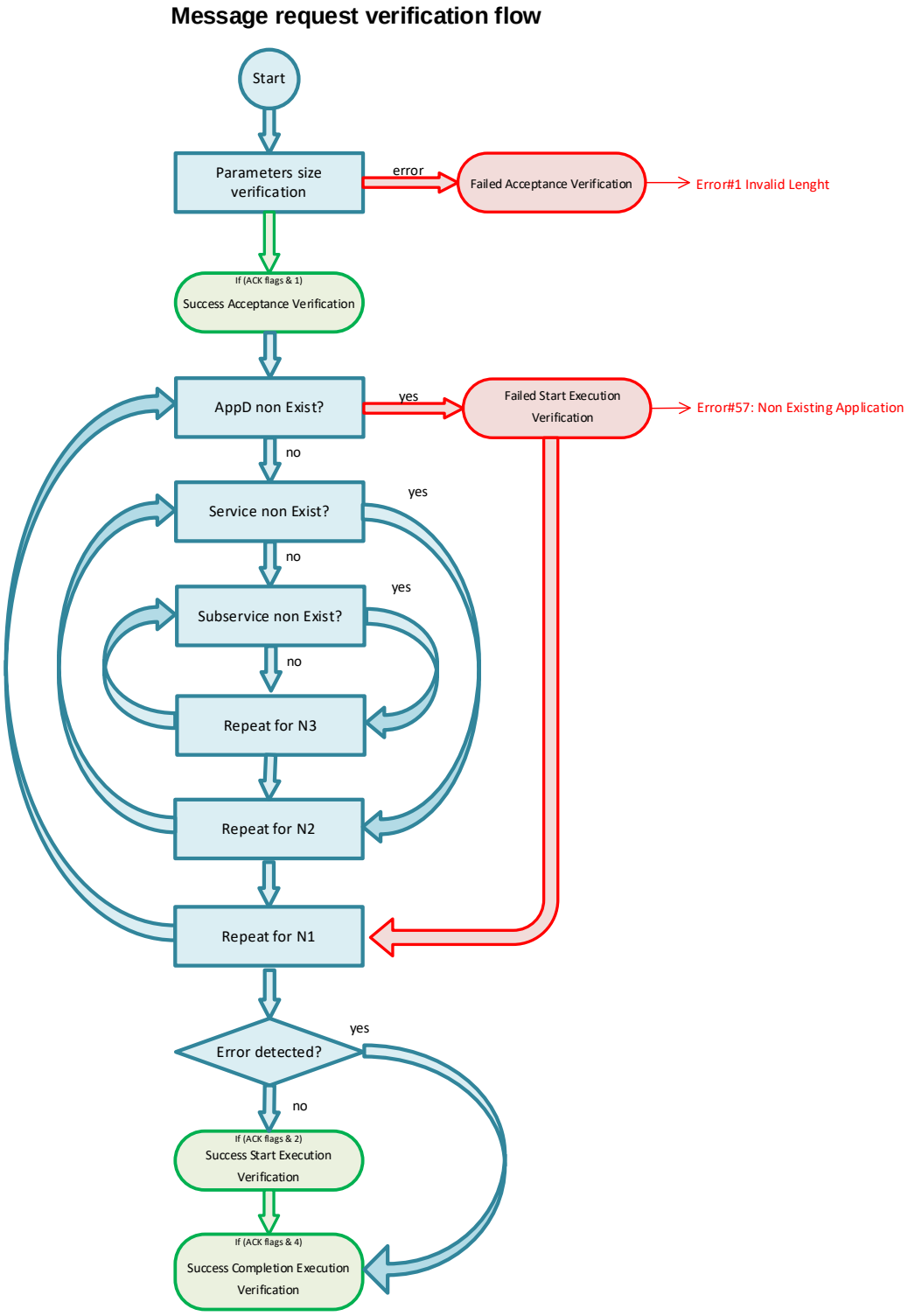
The json init file has next format:

```
{
    "filter": [
        {
            "apid": 25,
            "type": [
                {
                    "serviceType": 3,
                    "numSubType": 1,
                    "serviceSubType": [
                        10
                    ]
                }
            ]
        },
        {
            "apid": 3,
            "type": [
                {
                    "serviceType": 1,
                    "numSubType": 10,
                    "serviceSubType": [
                        1,
                        2,
                        3,
                        4,
                        5,
                        6,
                        7,
                        8,
                        9,
                        10
                    ]
                },
                {
                    "serviceType": 3,
                    "numSubType": 4,
                    "serviceSubType": [
                        1,
                        2,
                        5,
                        9
                    ]
                }
            ]
        }
    ]
```

Subtypes requests

TC[14,1] add report types to the application process forward-control configuration





TC[14,2] delete report types from the application process forward-control configuration

Delete report types from the application process forward-control configuration

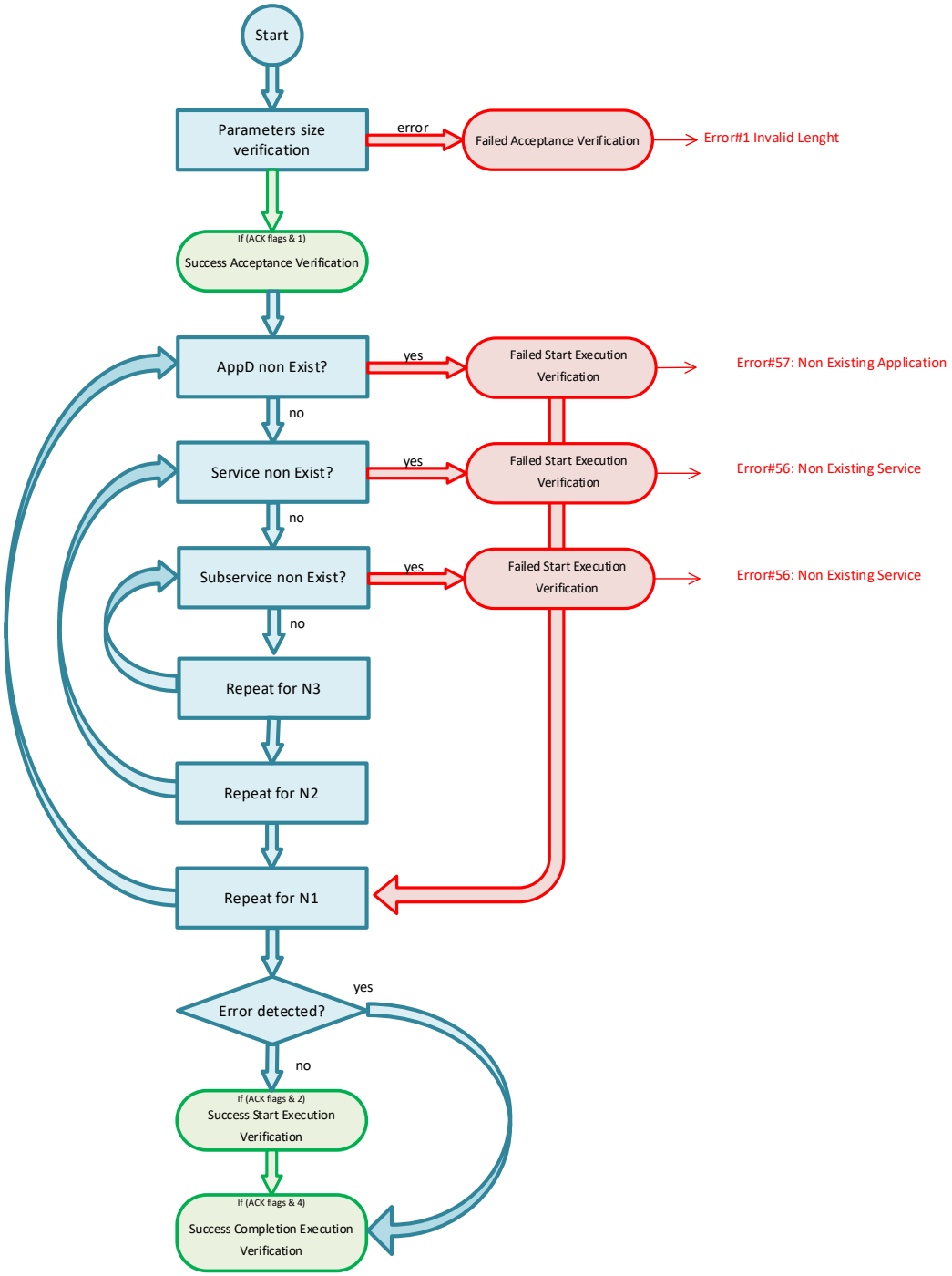
repeated N1 times

repeated N2 times

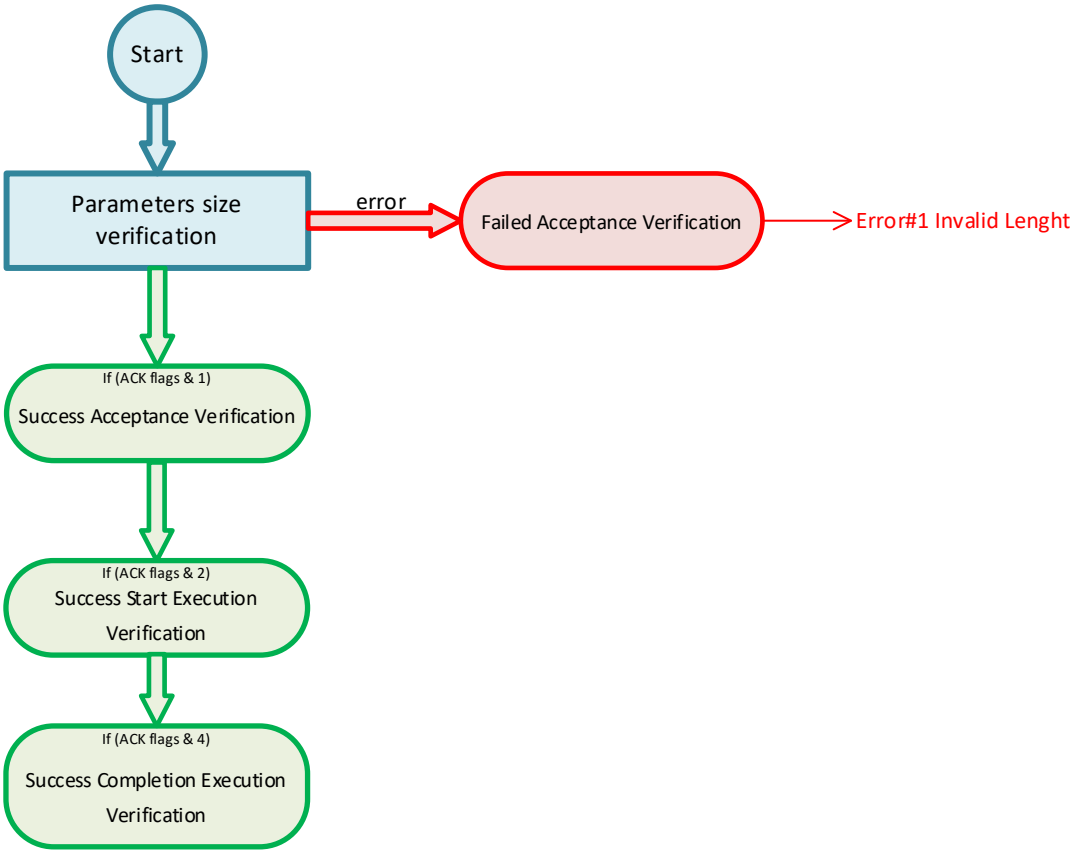
repeated N3 times

N1	application process ID	N2	service type	N3	message subtype
unsigned integer	enumerated	unsigned integer	enumerated	unsigned integer	enumerated

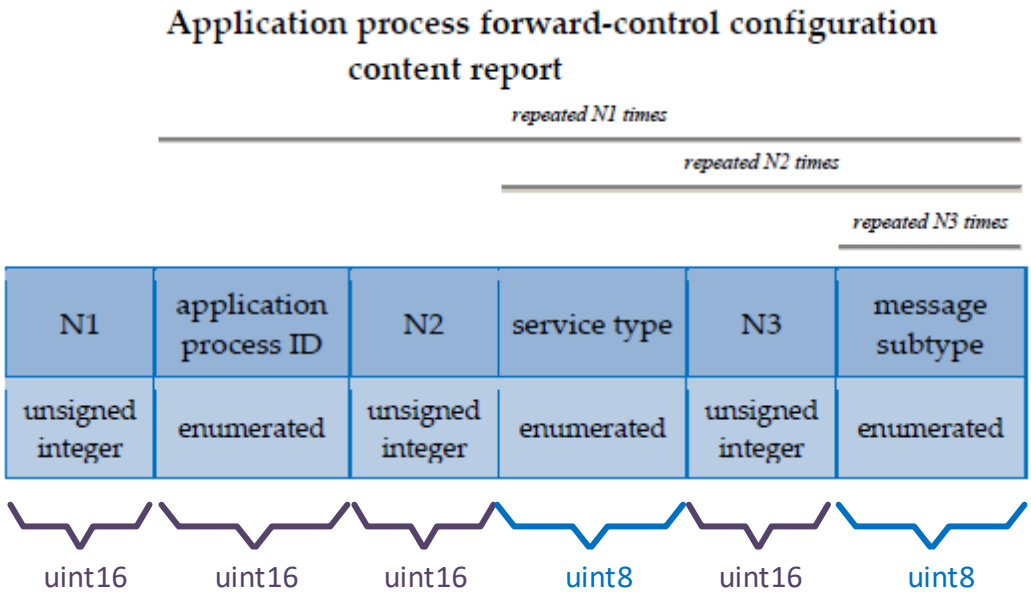
Message request verification flow



TC[14,3] report the content of the application process forward-control configuration

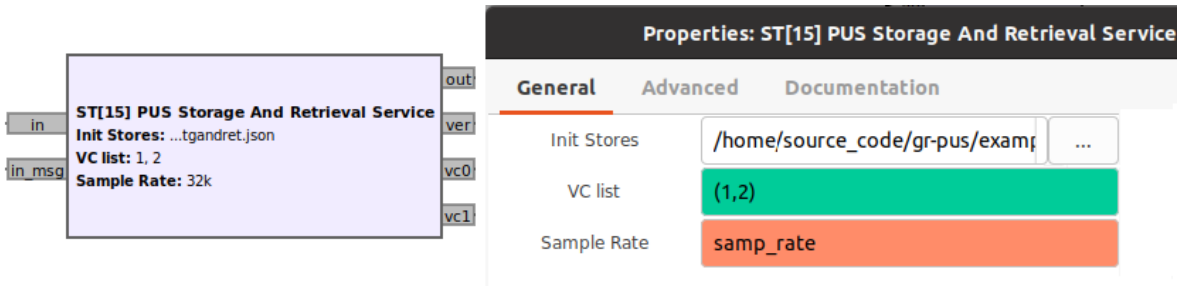


TM[14,4] application process forward-control configuration content report



2.16 [ST15]STORAGE AND RETRIEVAL

The ***ST[15] Storage and Retieval Service*** block will received all message request at its input port and if those request are for service ST[15] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Init Stores

Path to the json file with the start up store definitions, left empty if no init required

VC list

The list of configured Virtual Channel outputs for the stores, each VC in the list will match against the outputs VC0…VCn

Sample Rate

The “bit rate” (overall, see PUS implementation chapter) at whitch the stored messages will be send thru the Virtual Channel outputs

Messages

In

The message requests input

Out

The message report output

in_msg

The in_msg Input will receive the message reports, any reports matching the definitions will be stored

VC0…n

Outputs matching the VC list for message retrieval, each time a packet store mesagge is in progress it will send the stored messages through these outputs according to the packet store configuration

The json init file has next format:

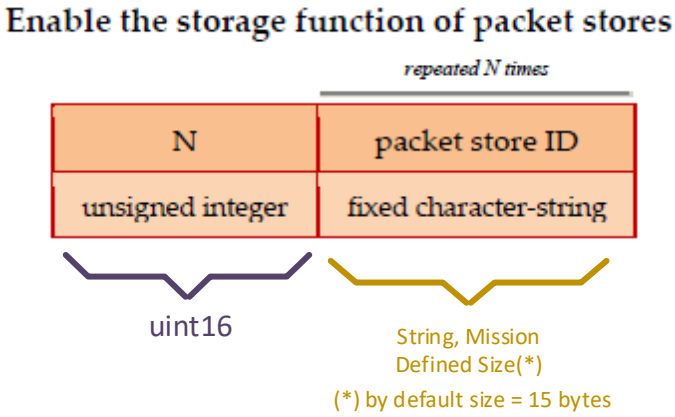
```
{
    "numAppID": 3,
    "appIDMon": [
        3,
        25,
        30
    ],
    "store": [
        {
            "name": "housekeeping",
            "packetStoreSize": 512,
            "packetStoreType": 0,
            "storageStatus": false,
            "virtualChannel": 1,
            "filter": [
                {
                    "apid": 25,
                    "type": [
                        {
                            "serviceType": 3,
                            "numSubType": 1,
                            "serviceSubType": [
                                10
                            ]
                        },
                        {
                            "serviceType": 12,
                            "numSubType": 2,
                            "serviceSubType": [
                                10,
                                11
                            ]
                        }
                    ]
                },
                {
                    "apid": 3,
                    "type": [
                        {
                            "serviceType": 1,
                            "numSubType": 10,
                            "serviceSubType": [
                                1,
                                2,
                                3,
```

The appIDMon will included all AppID that shall be monitored

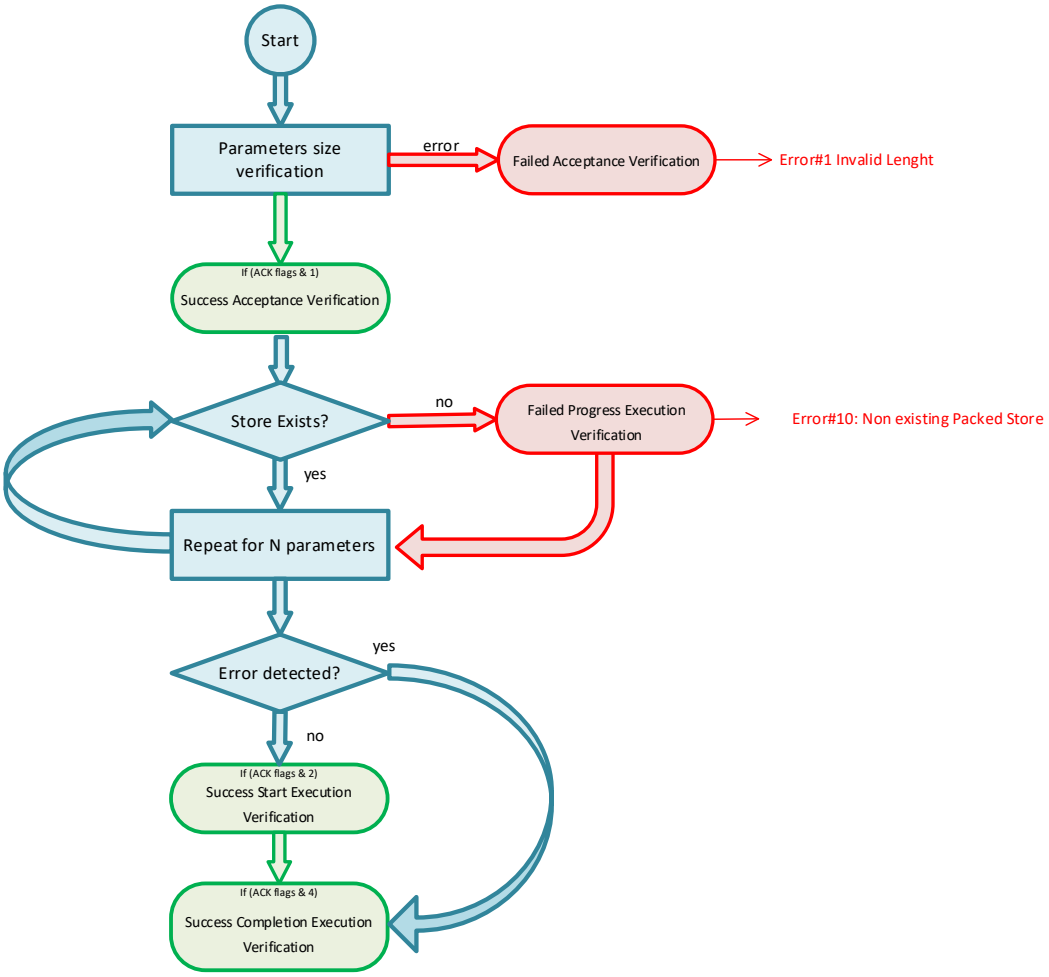
PacketStoreType are Circular = 0 or Bounded = 1
StorageStatus if it is enabled=true or disable=false

Subtypes requests

TC[15,1] enable the storage function of packet stores



Message request verification flow



TC[15,2] disable the storage function of packet stores

Disable the storage function of packet stores

repeated N times

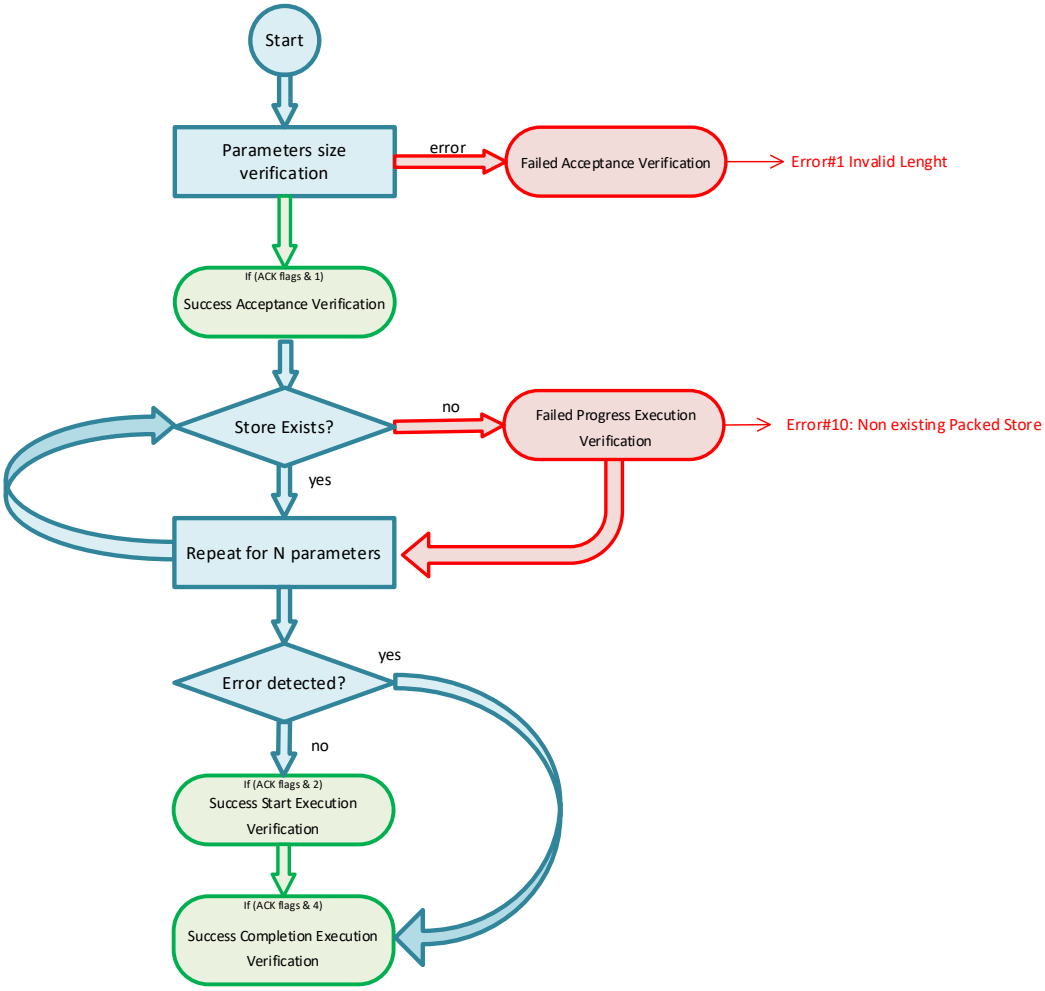
N	packet store ID
unsigned integer	fixed character-string

uint16

String, Mission Defined Size(*)

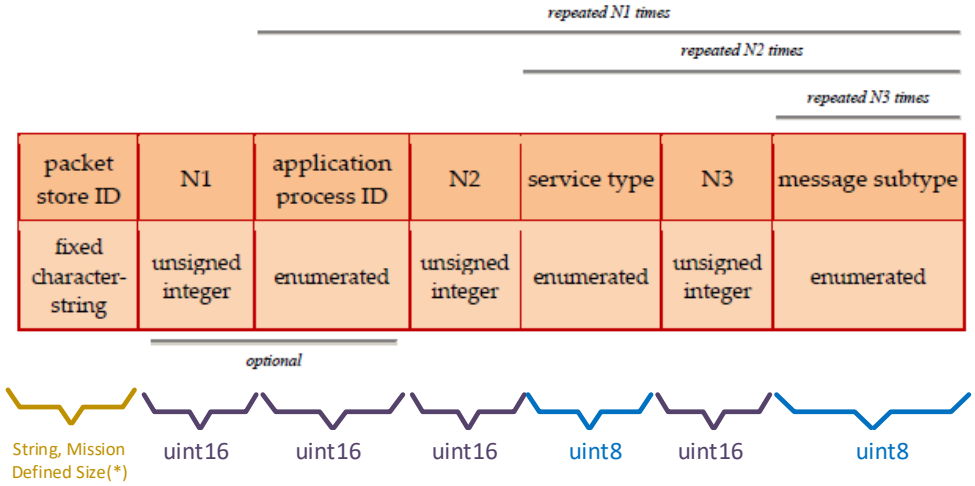
(*) by default size = 15 bytes

Message request verification flow



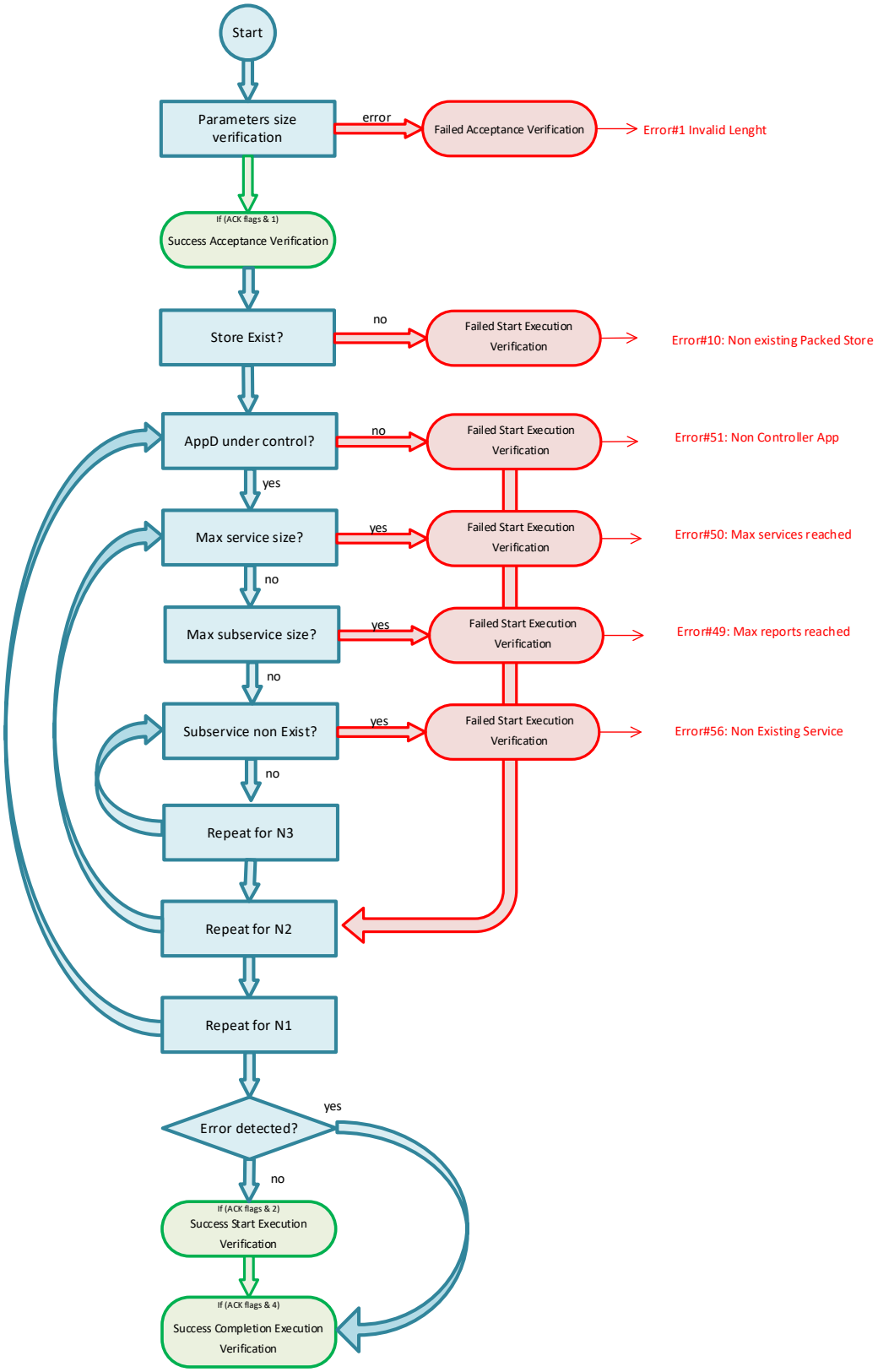
TC[15,3] add report types to the application process storage-control configuration

Add report types to the application process storage-control configuration

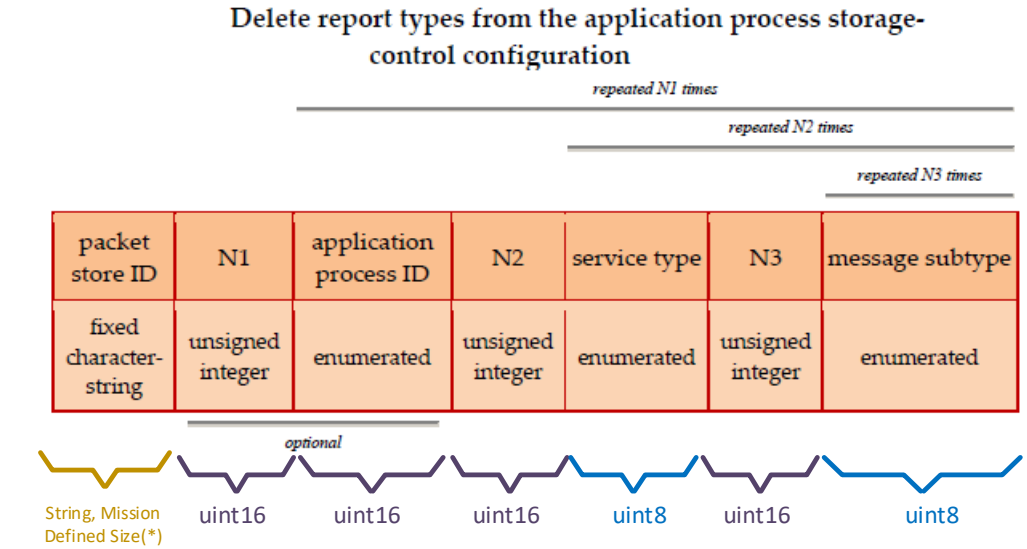


(*) by default size = 15 bytes

Message request verification flow

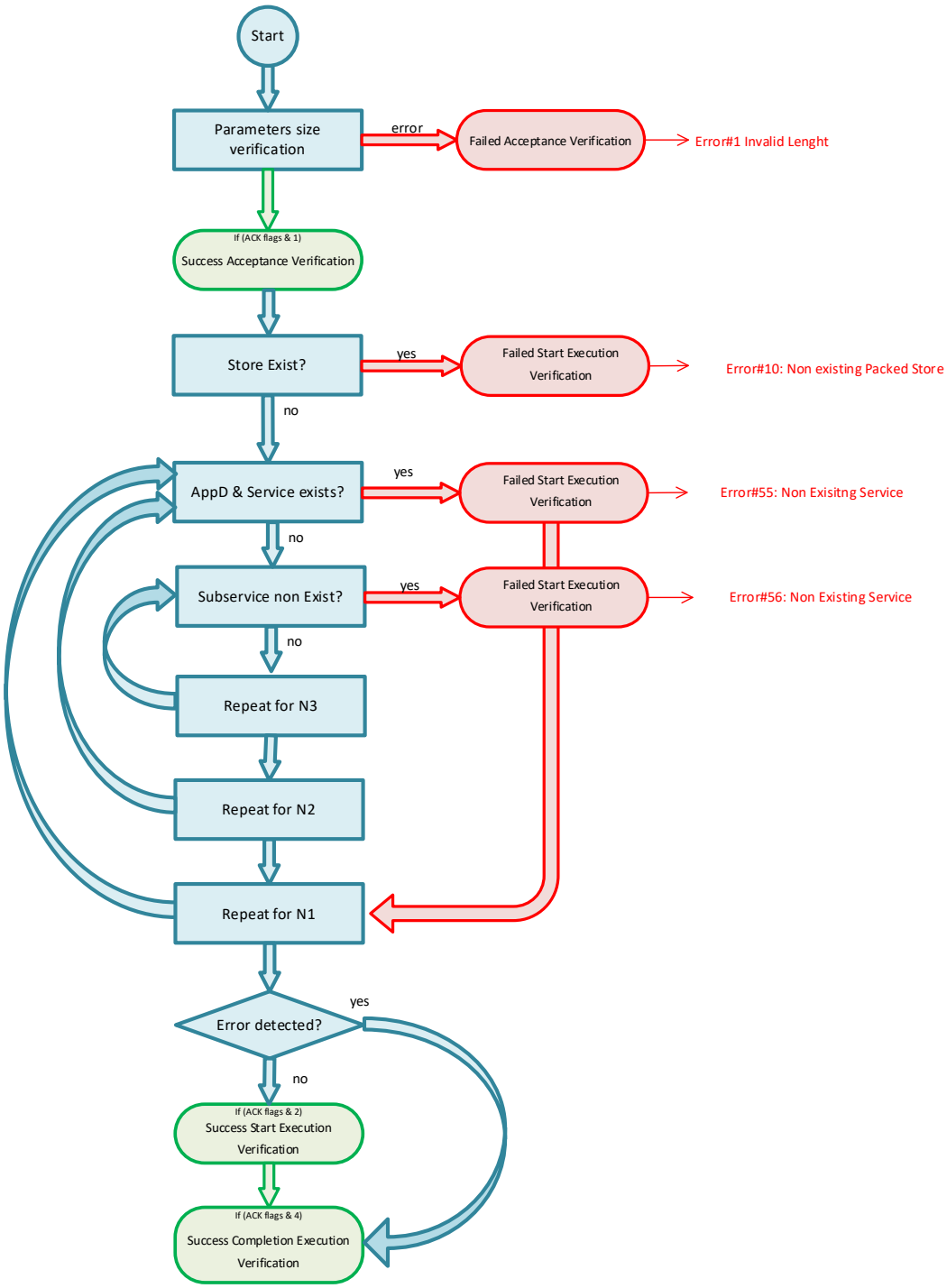


TC[15,4] delete report types from the application process storage-control configuration



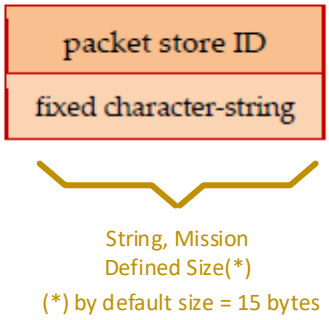
(*) by default size = 15 bytes

Message request verification flow

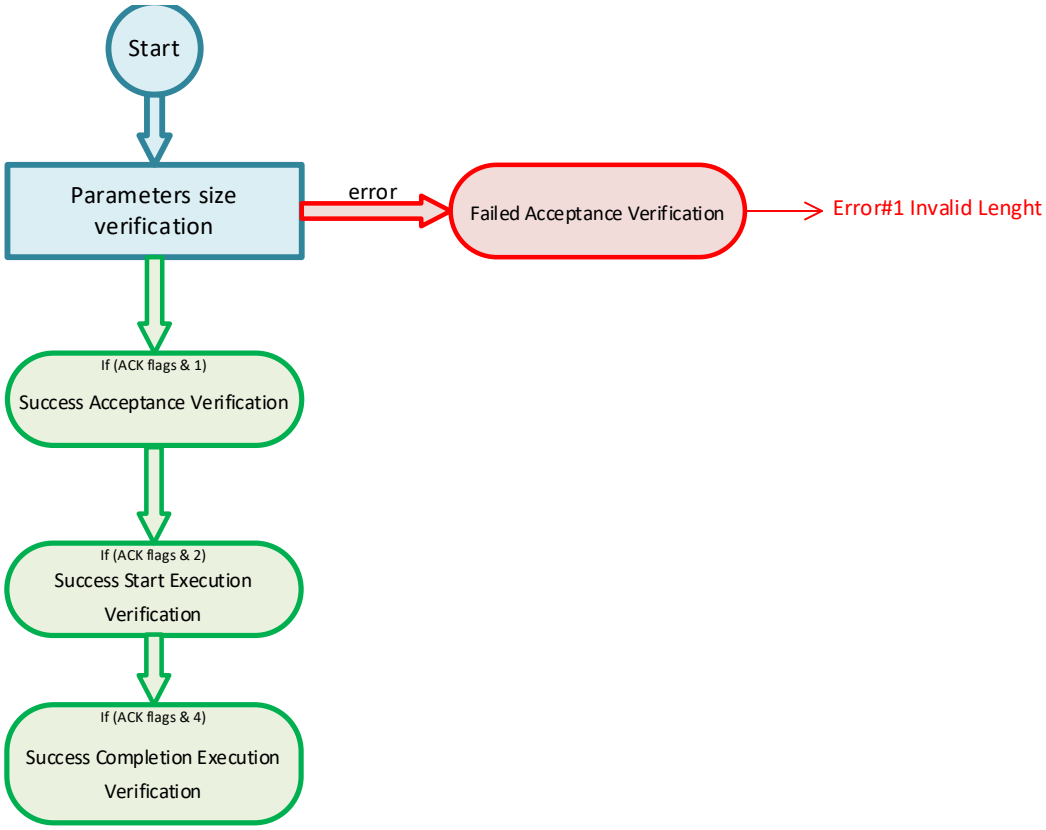


TC[15,5] report the content of the application process storage-control configuration

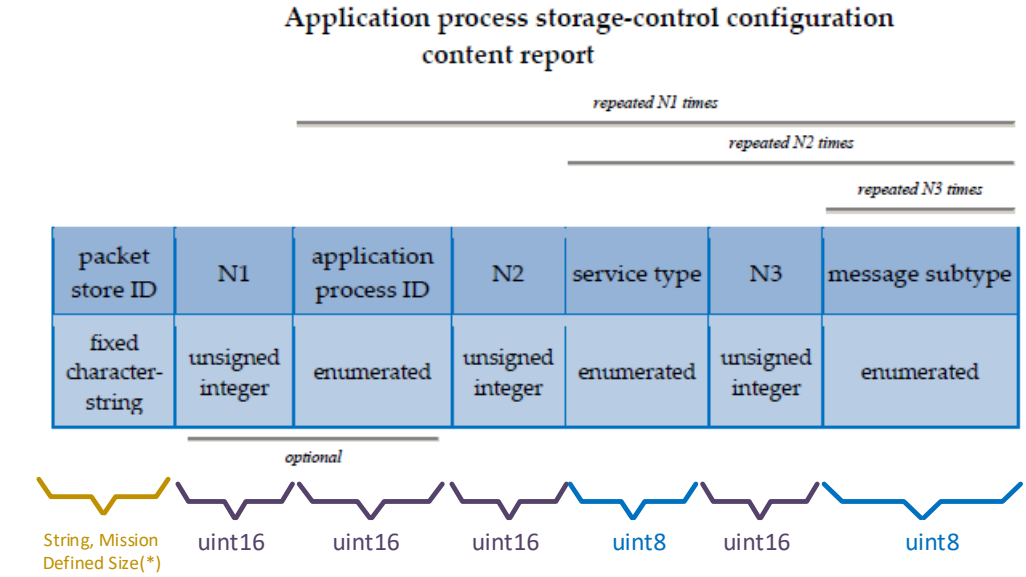
Report the content of the application process storage-control configuration



Message request verification flow

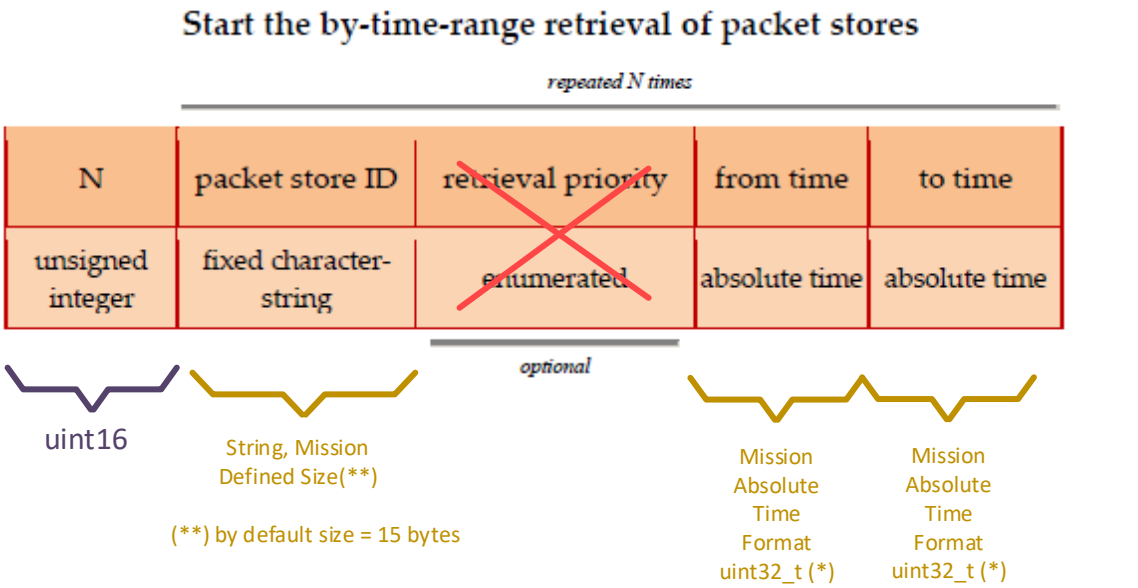


TM[15,6] application process storage-control configuration content report



(*) by default size = 15 bytes

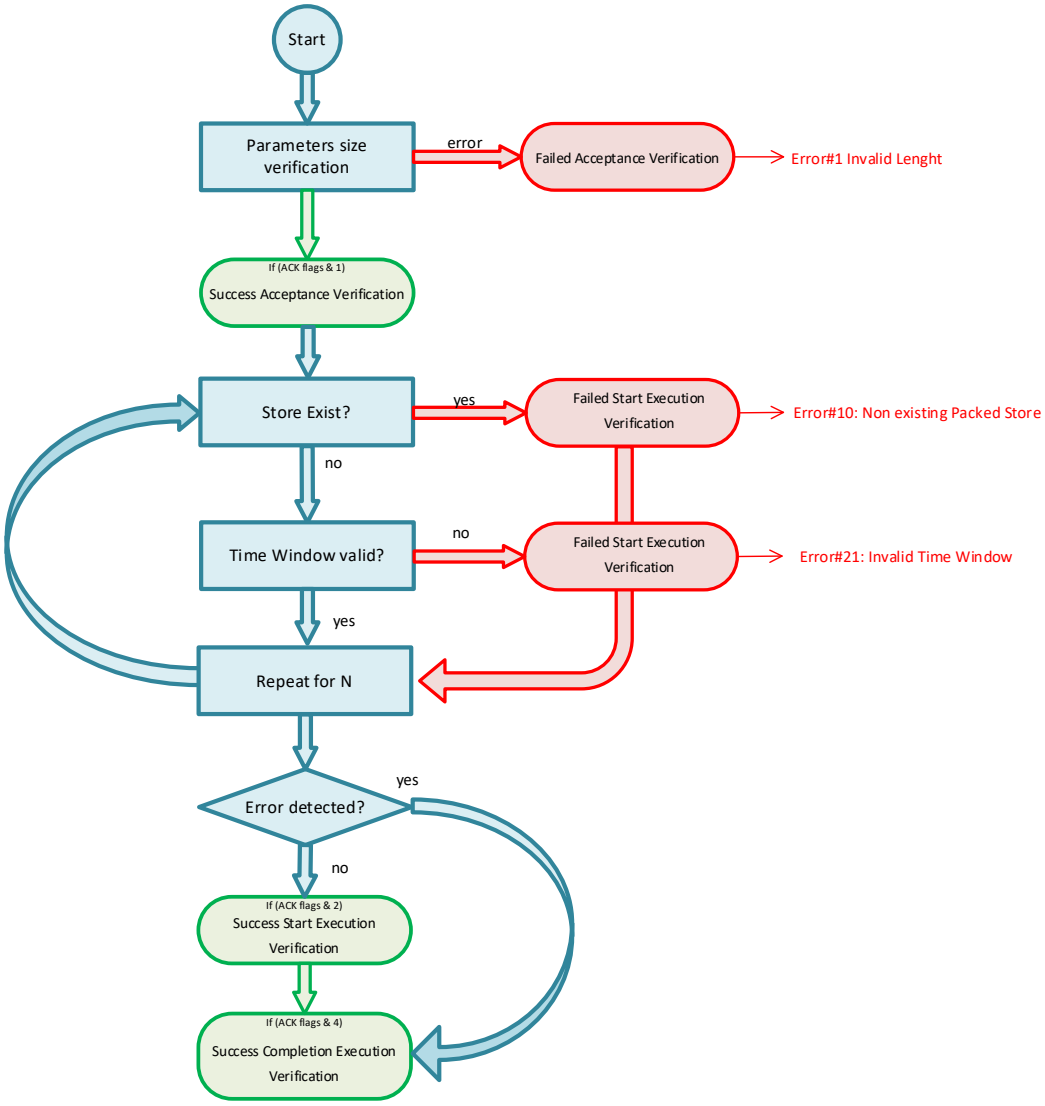
TC[15,9] start the by-time-range retrieval of packet stores



(**) by default size = 15 bytes

(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

Message request verification flow



TC[15,11] delete the content of packet stores up to the specified time

Delete the content of packet stores up to the specified time

repeated N times

storage time	N	packet store ID
absolute time	unsigned integer	fixed character-string

Mission Absolute Time Format uint32_t (*)

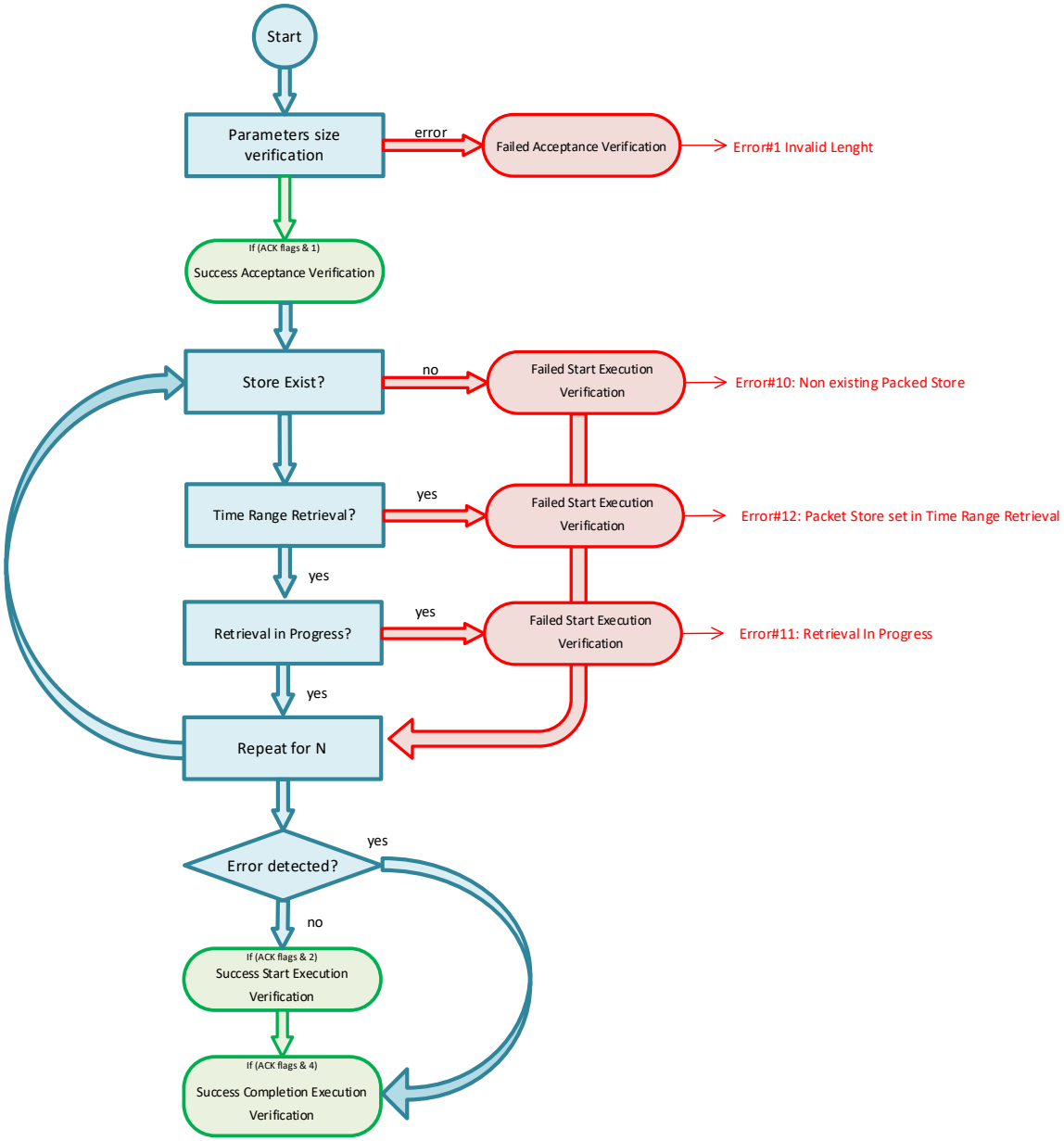
uint16

String, Mission Defined Size(**)

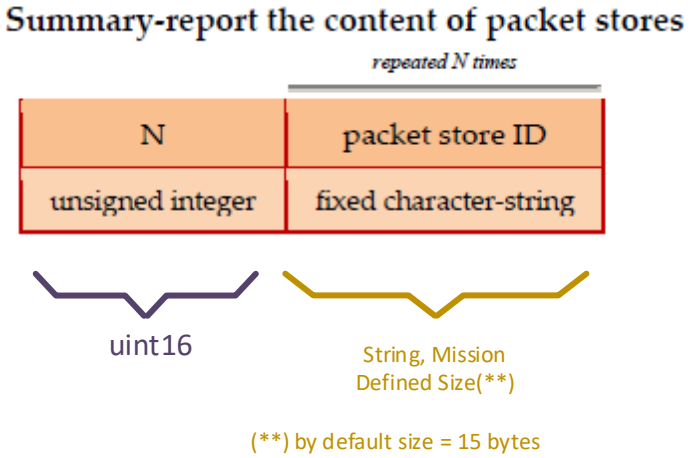
(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

(**) by default size = 15 bytes

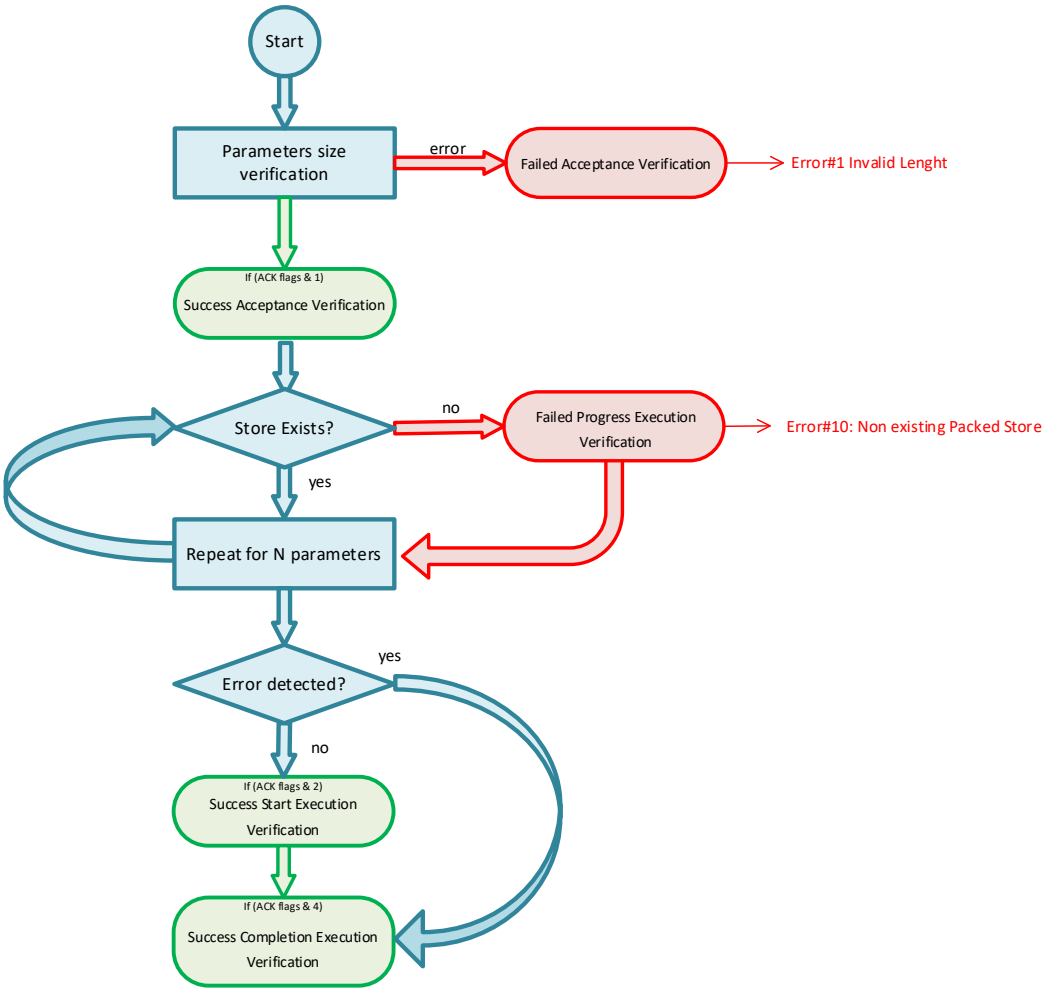
Message request verification flow



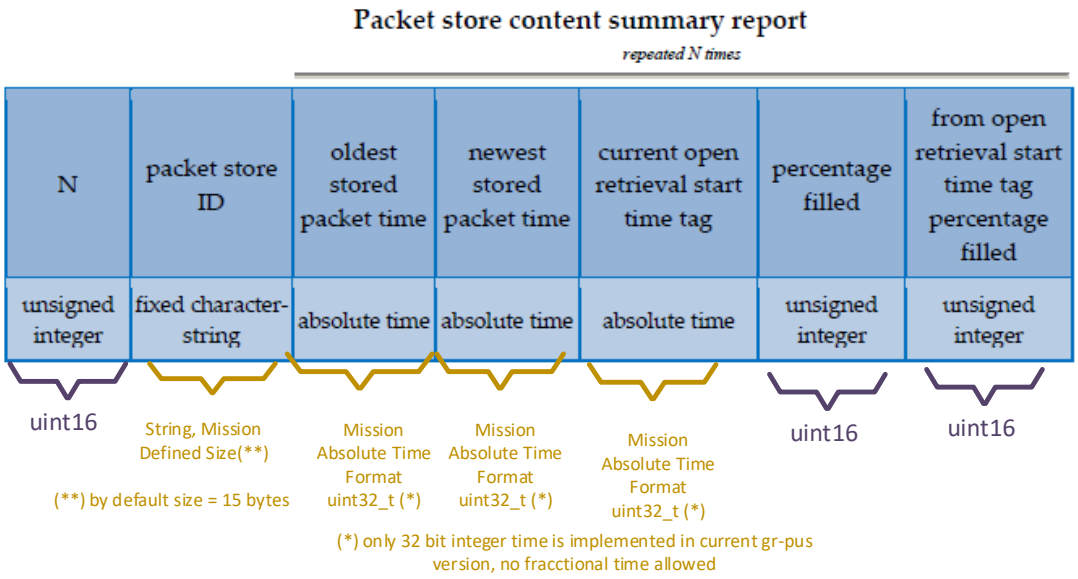
TC[15,12] summary-report the content of packet stores



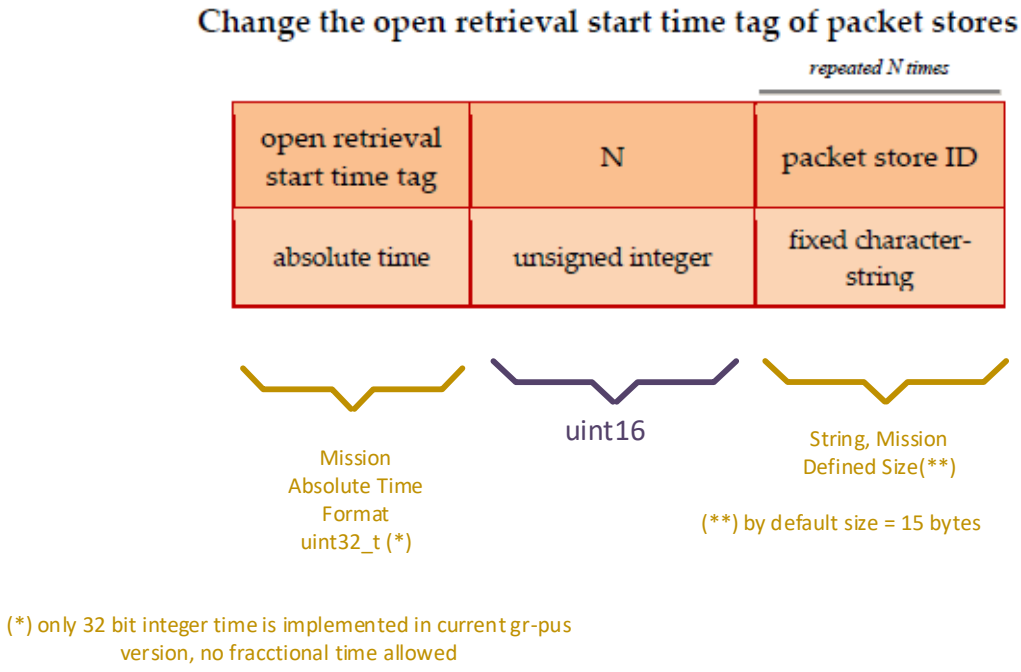
Message request verification flow



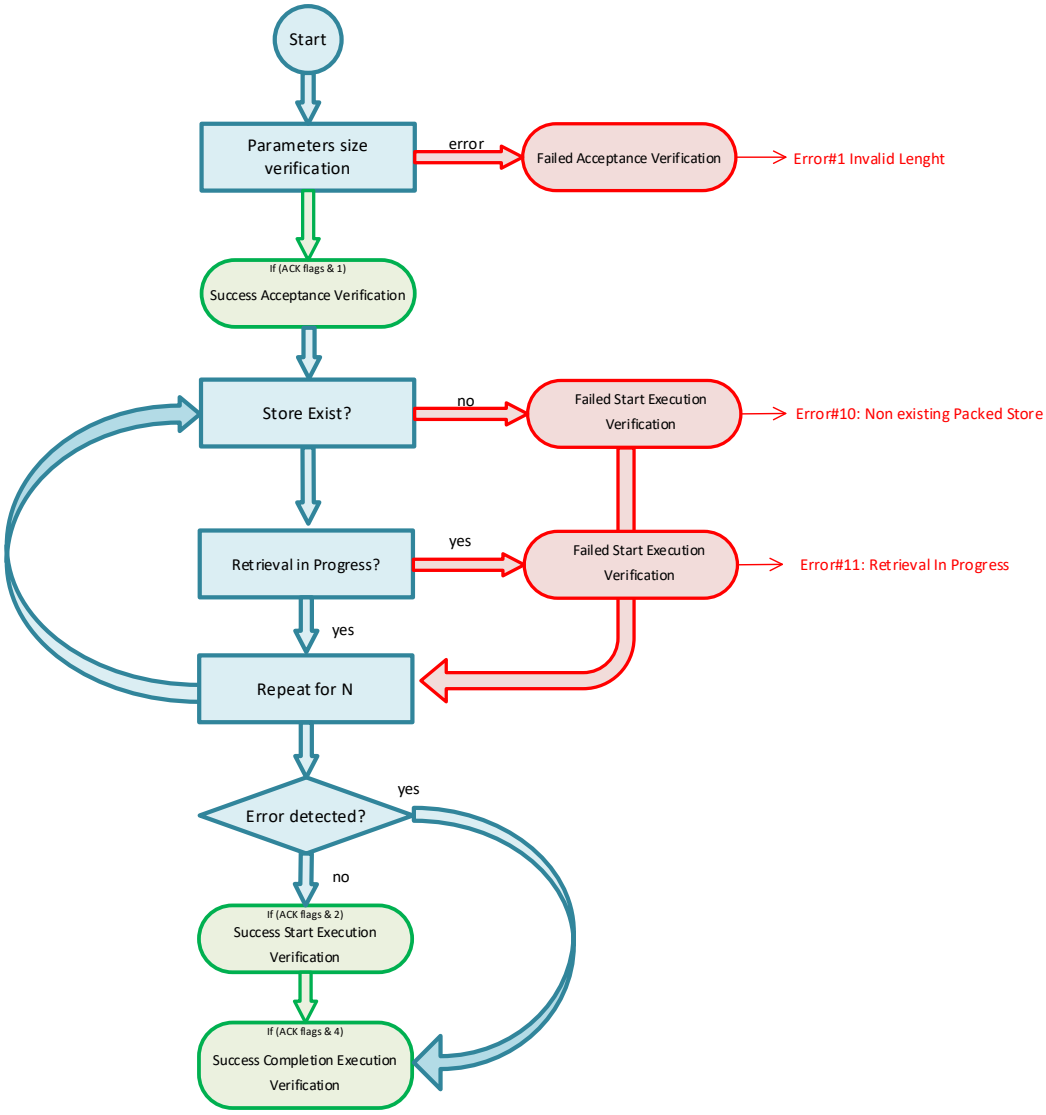
TM[15,13] packet store content summary report



TC[15,14] change the open retrieval start time tag of packet stores



Message request verification flow



TC[15,15] resume the open retrieval of packet stores

Resume the open retrieval of packet stores

repeated N times

N	packet store ID	retrieval priority
unsigned integer	fixed character-string	enumerated

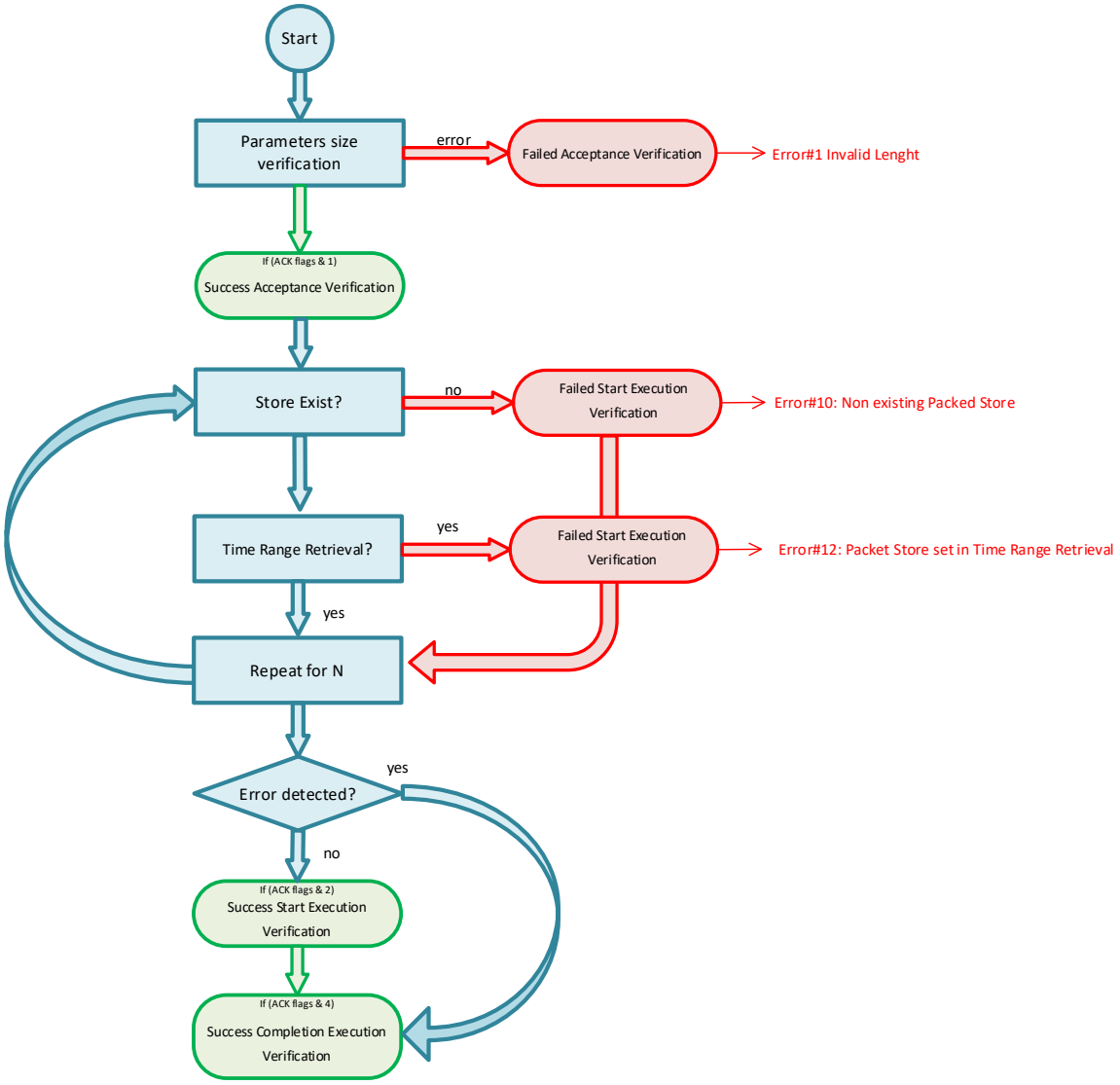
optional

uint16

String, Mission Defined Size(**)

(**) by default size = 15 bytes

Message request verification flow



TC[15,16] suspend the open retrieval of packet stores

Suspend the open retrieval of packet stores

repeated N times

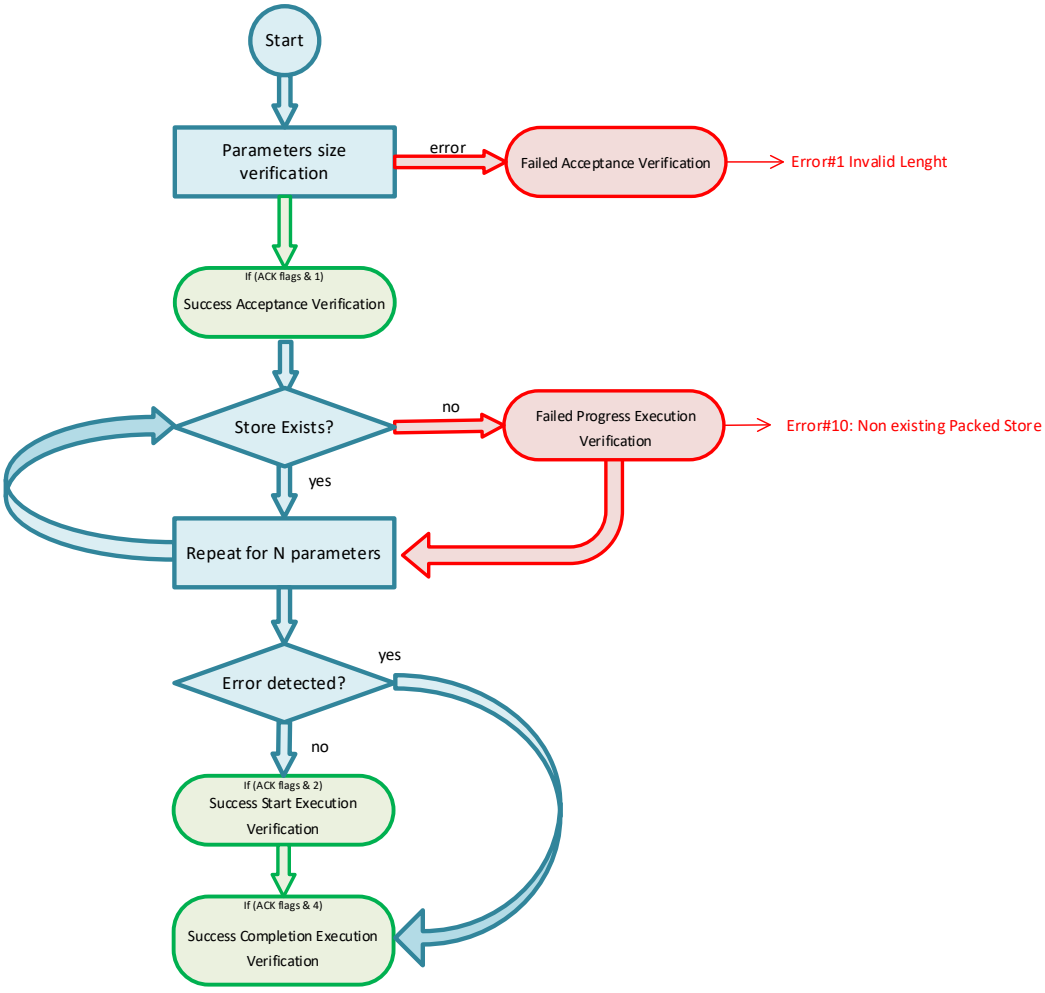
N	packet store ID
unsigned integer	fixed character-string

uint16

String, Mission Defined Size(**)

(**) by default size = 15 bytes

Message request verification flow



TC[15,17] abort the by-time-range retrieval of packet stores

Abort the by-time-range retrieval of packet stores

repeated N times

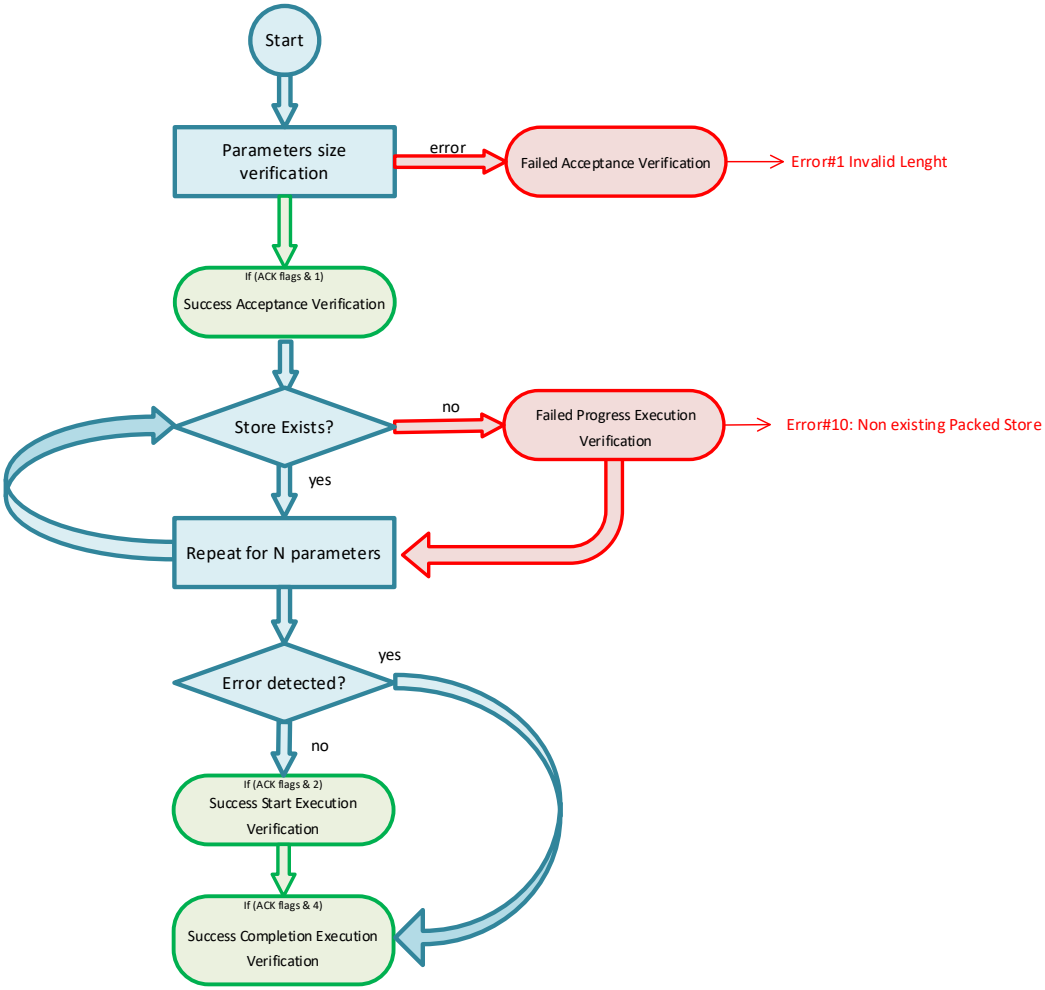
N	packet store ID
unsigned integer	fixed character-string

uint16

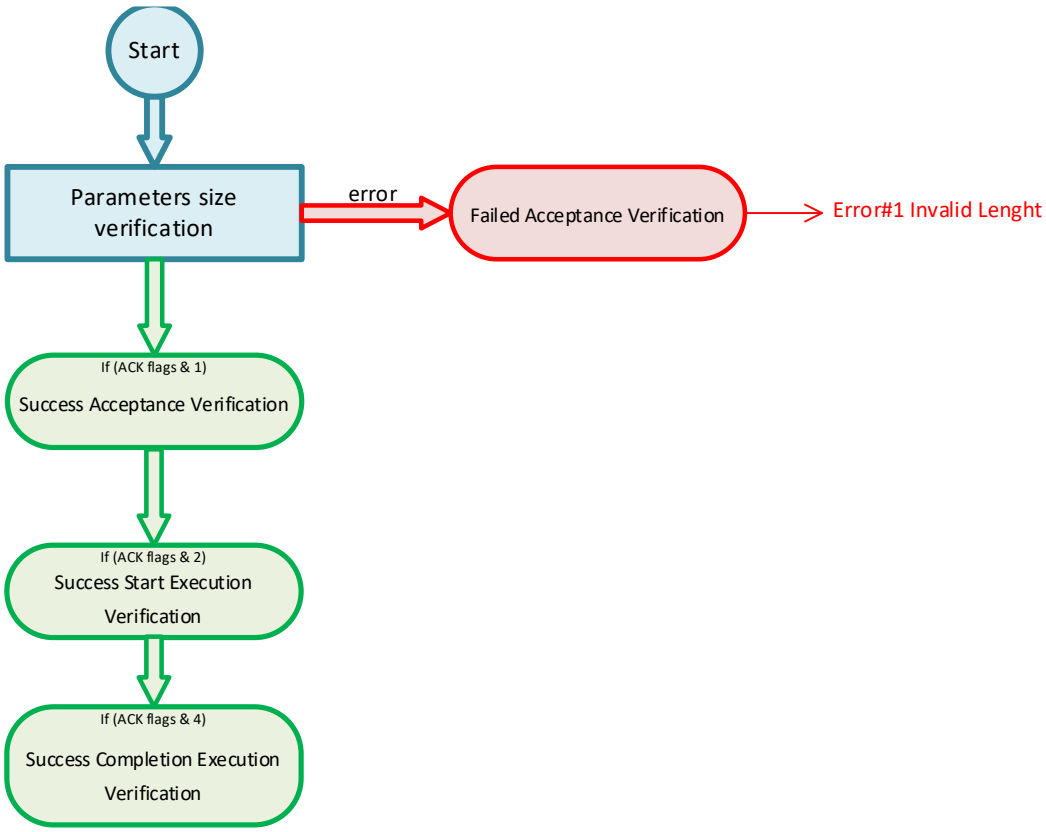
String, Mission Defined Size(**)

(**) by default size = 15 bytes

Message request verification flow



TC[15,18] report the status of each packet store



TM[15,19] packet store status report

Packet store status report

repeated N times

N	packet store ID	packet store status	packet store open retrieval status	packet store by-time-range retrieval status
unsigned integer	fixed character-string	enumerated	enumerated	enumerated

optional

(**) by default size = 15 bytes

TC[15,20] create packet stores

Create packet stores

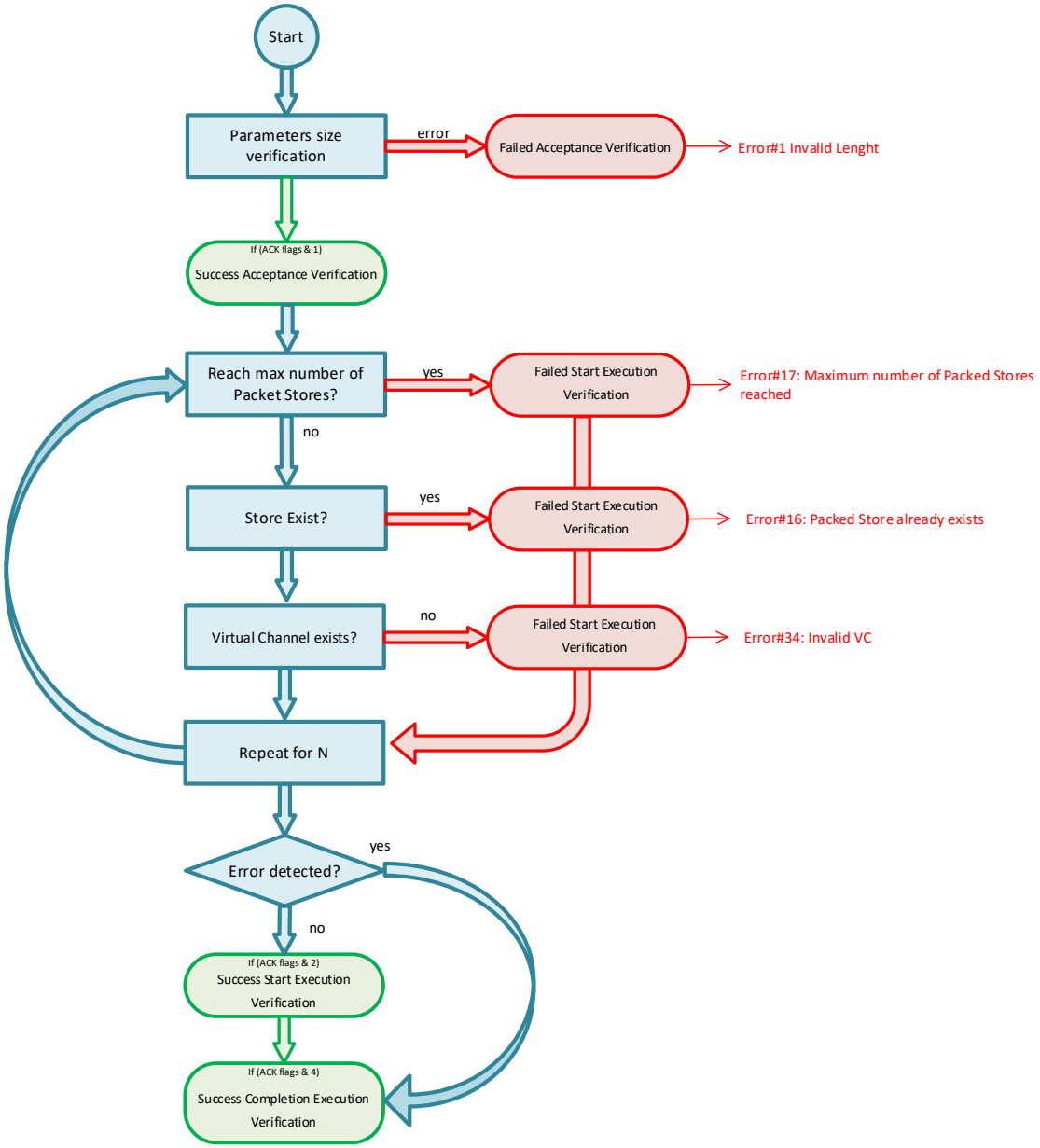
repeated N times

N	packet store ID	packet store size	packet store type	packet store virtual channel
unsigned integer	fixed character-string	unsigned integer	enumerated	enumerated

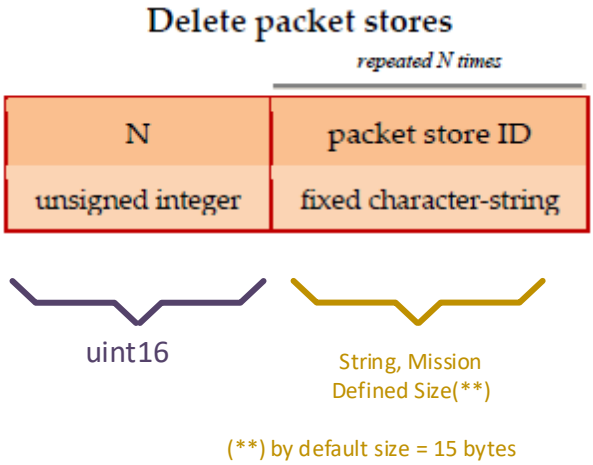
optional *optional*

(**) by default size = 15 bytes

Message request verification flow

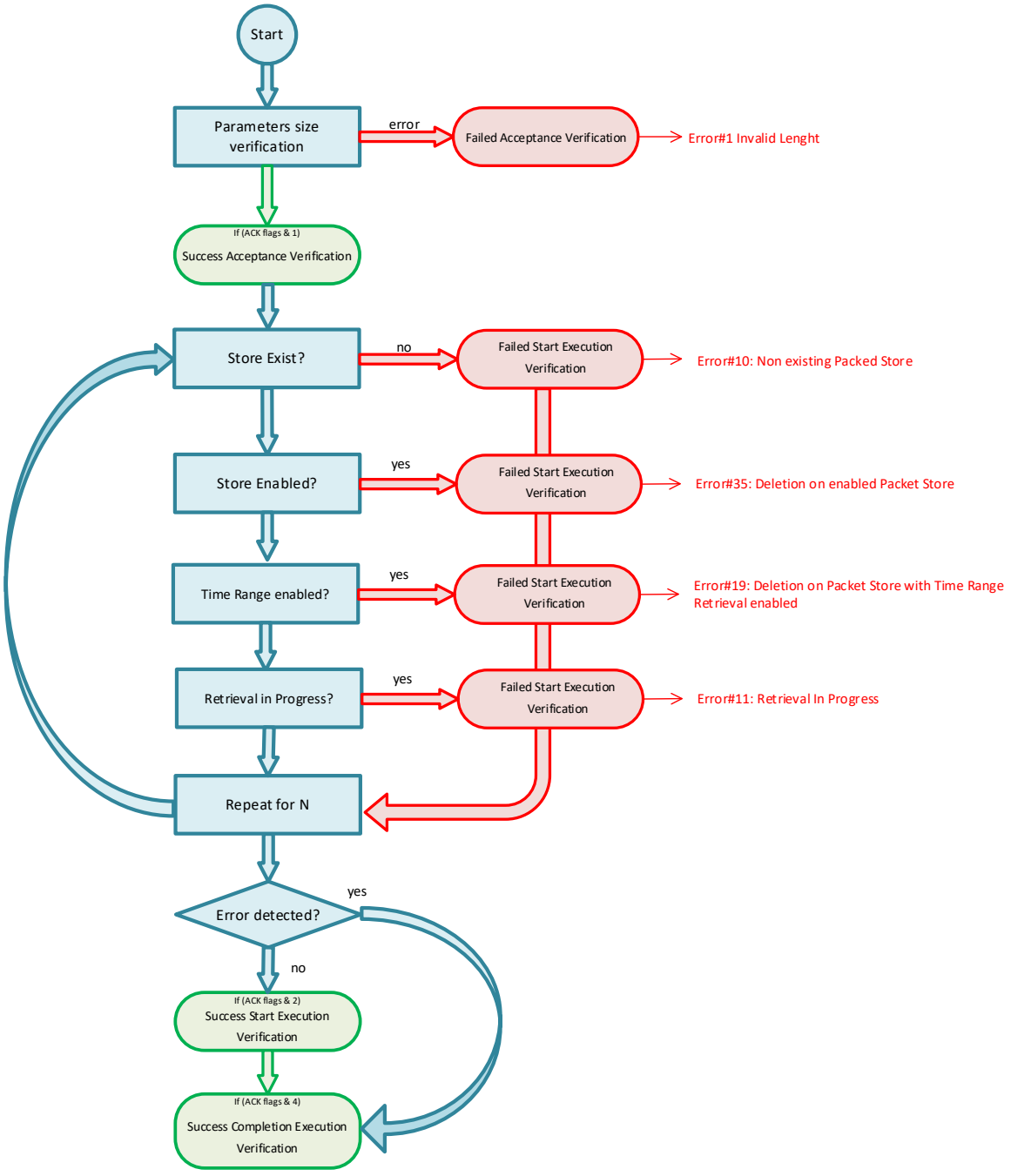


TC[15,21] delete packet stores



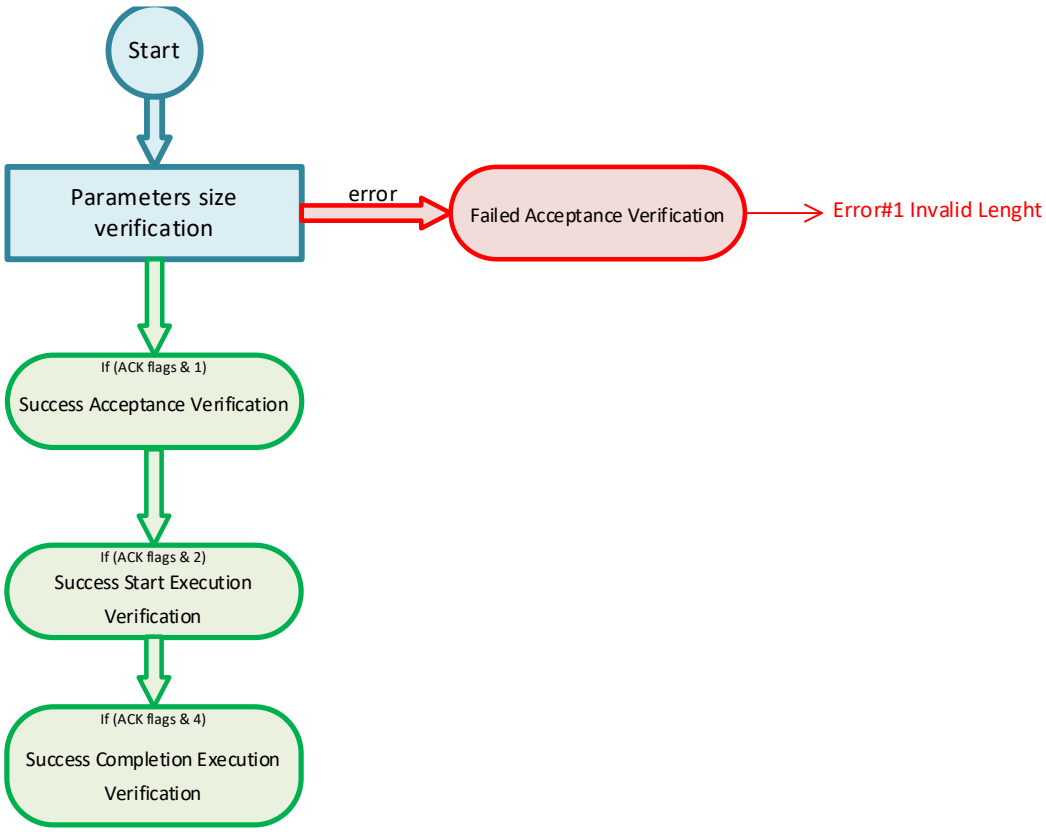
(**) by default size = 15 bytes

Message request verification flow



TC[15,22] report the configuration of each packet store

Message request verification flow



TM[15,23] packet store configuration report

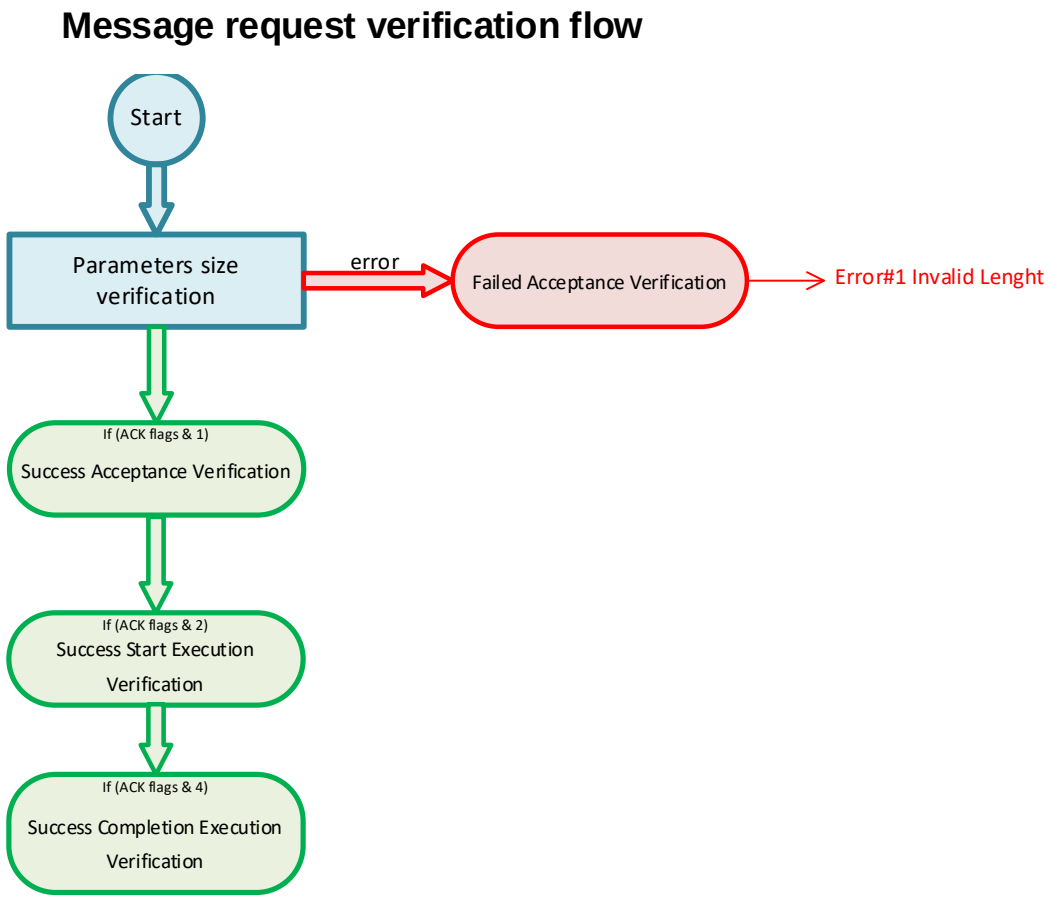
Packet store configuration report

repeated N times

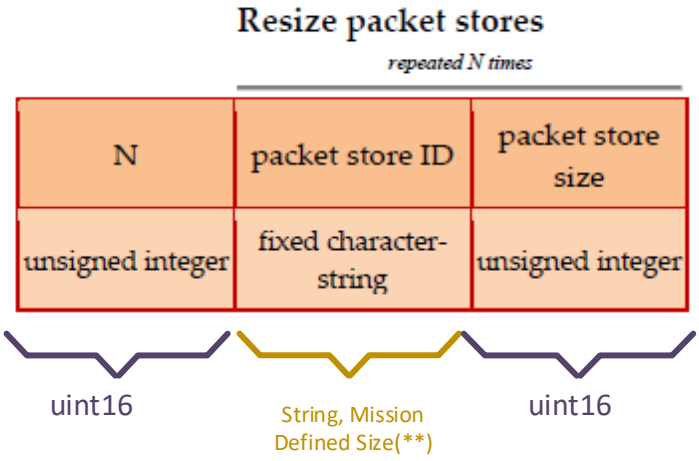
N	packet store ID	packet store size	packet store type	packet store virtual channel
unsigned integer	fixed character-string	unsigned integer	enumerated	enumerated
			optional	*optional*
uint16	String, Mission Defined Size(**)	uint16	uint8	uint8

(**) by default size = 15 bytes

TC[15,24] copy the packets contained in a packet store selected by time window

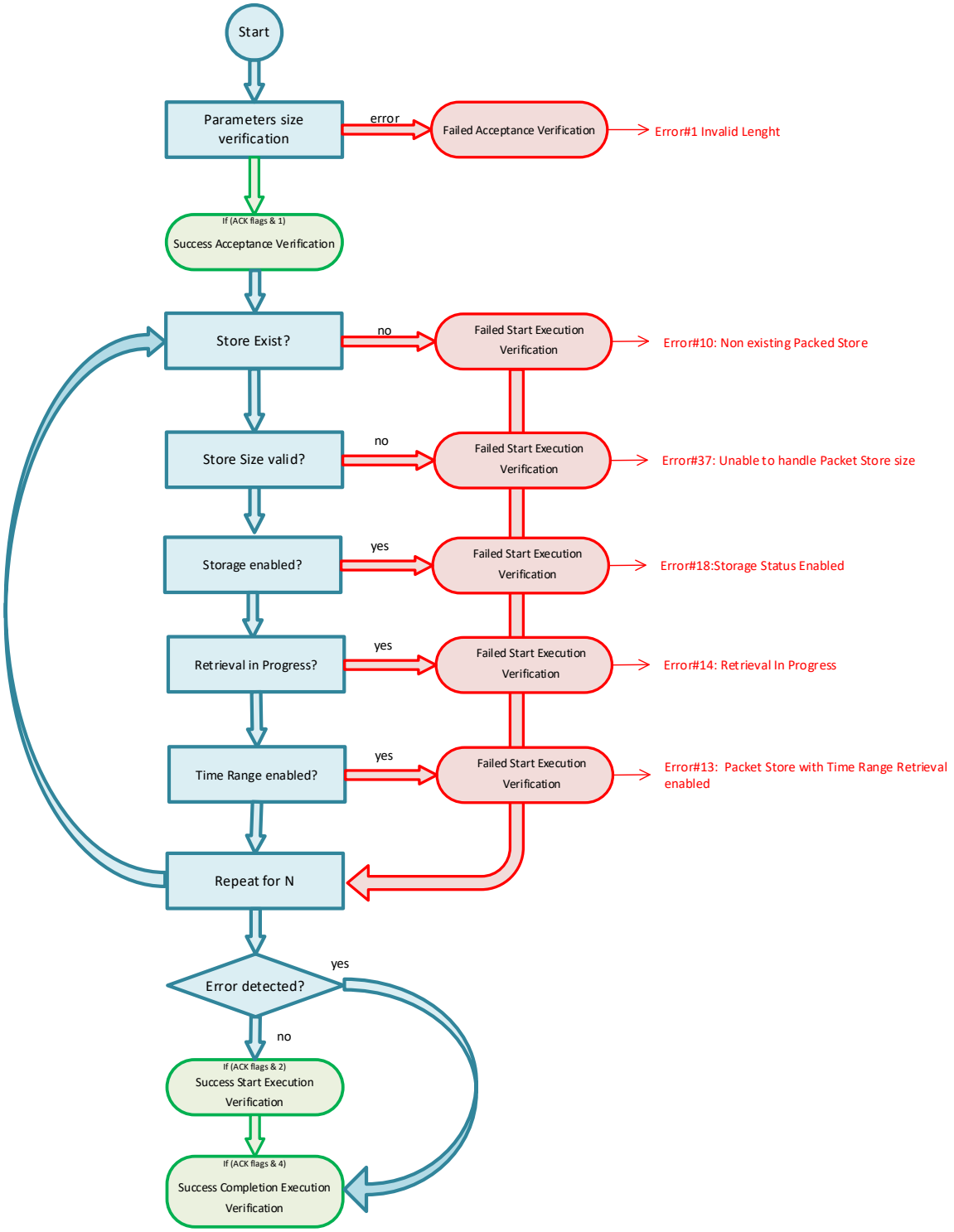


TC[15,25] resize packet stores



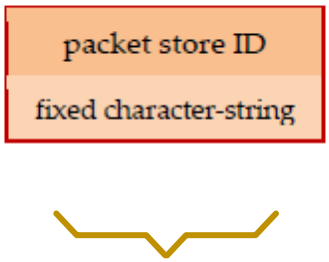
(**) by default size = 15 bytes

Message request verification flow



TC[15,26] change a packet store type to circular

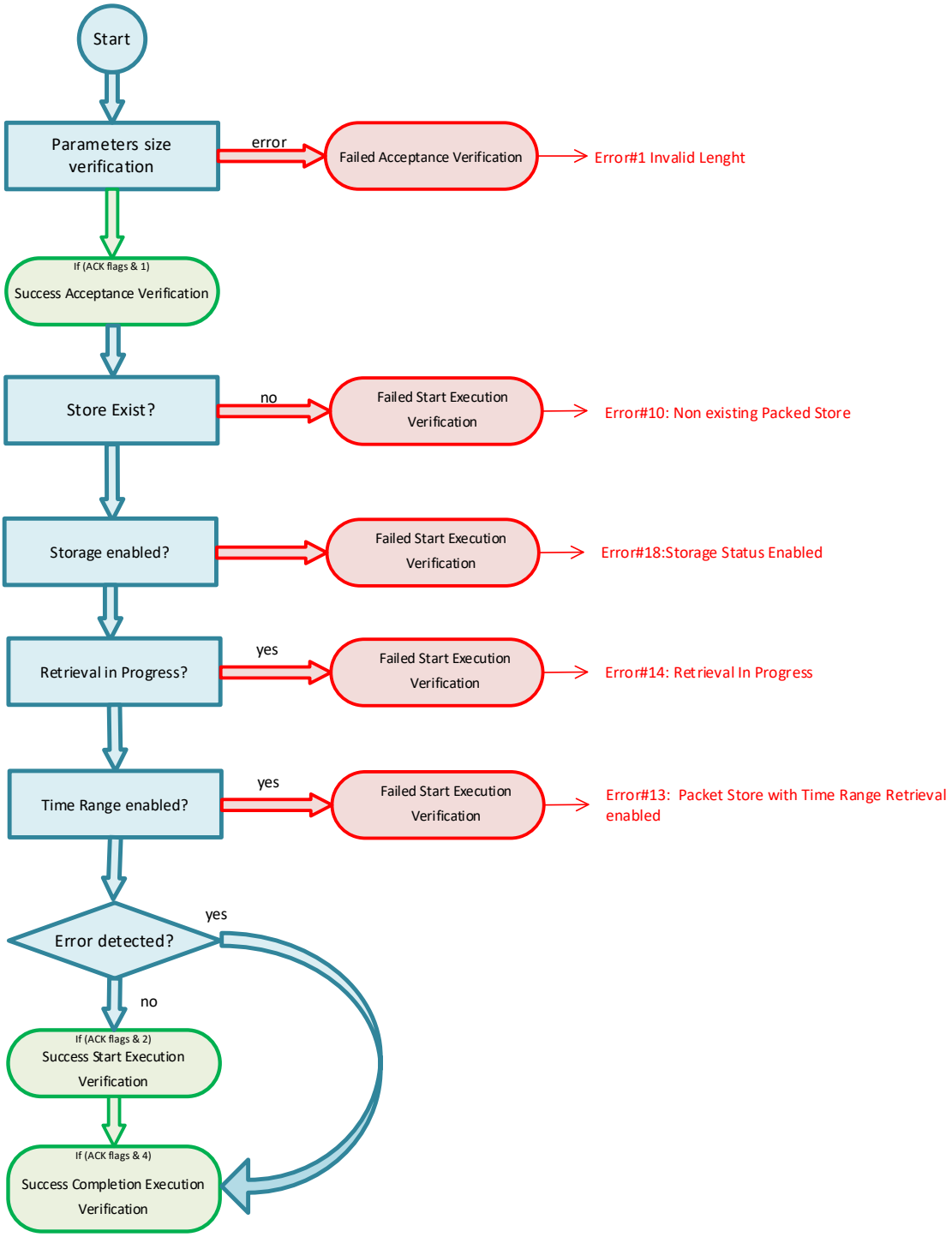
Change a packet store type to circular



String, Mission
Defined Size(**)

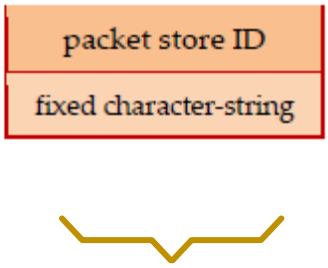
(**) by default size = 15 bytes

Message request verification flow



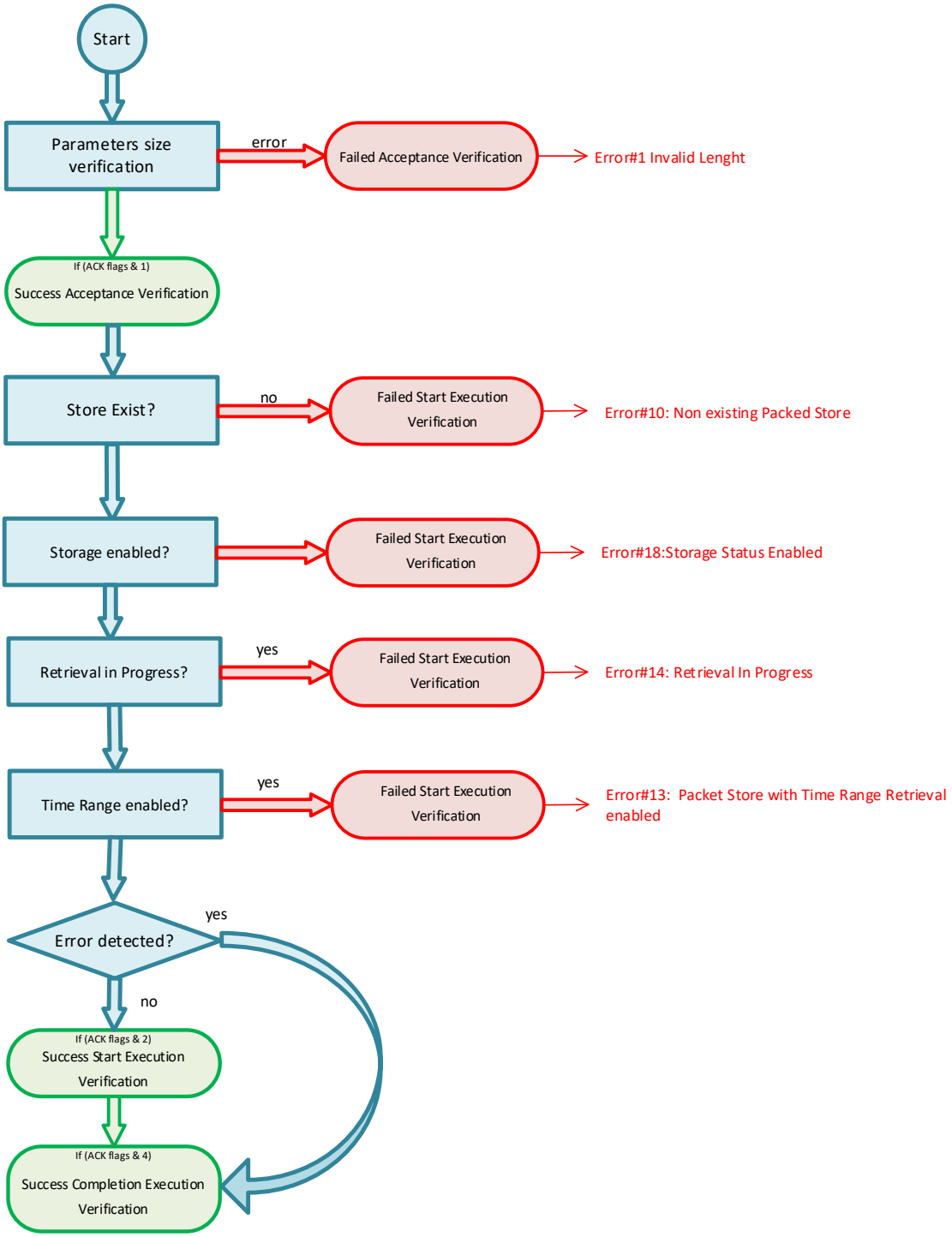
TC[15,27] change a packet store type to bounded

Change a packet store type to bounded



(**) by default size = 15 bytes

Message request verification flow



TC[15,28] change the virtual channel used by a packet store

Change the virtual channel used by a packet store

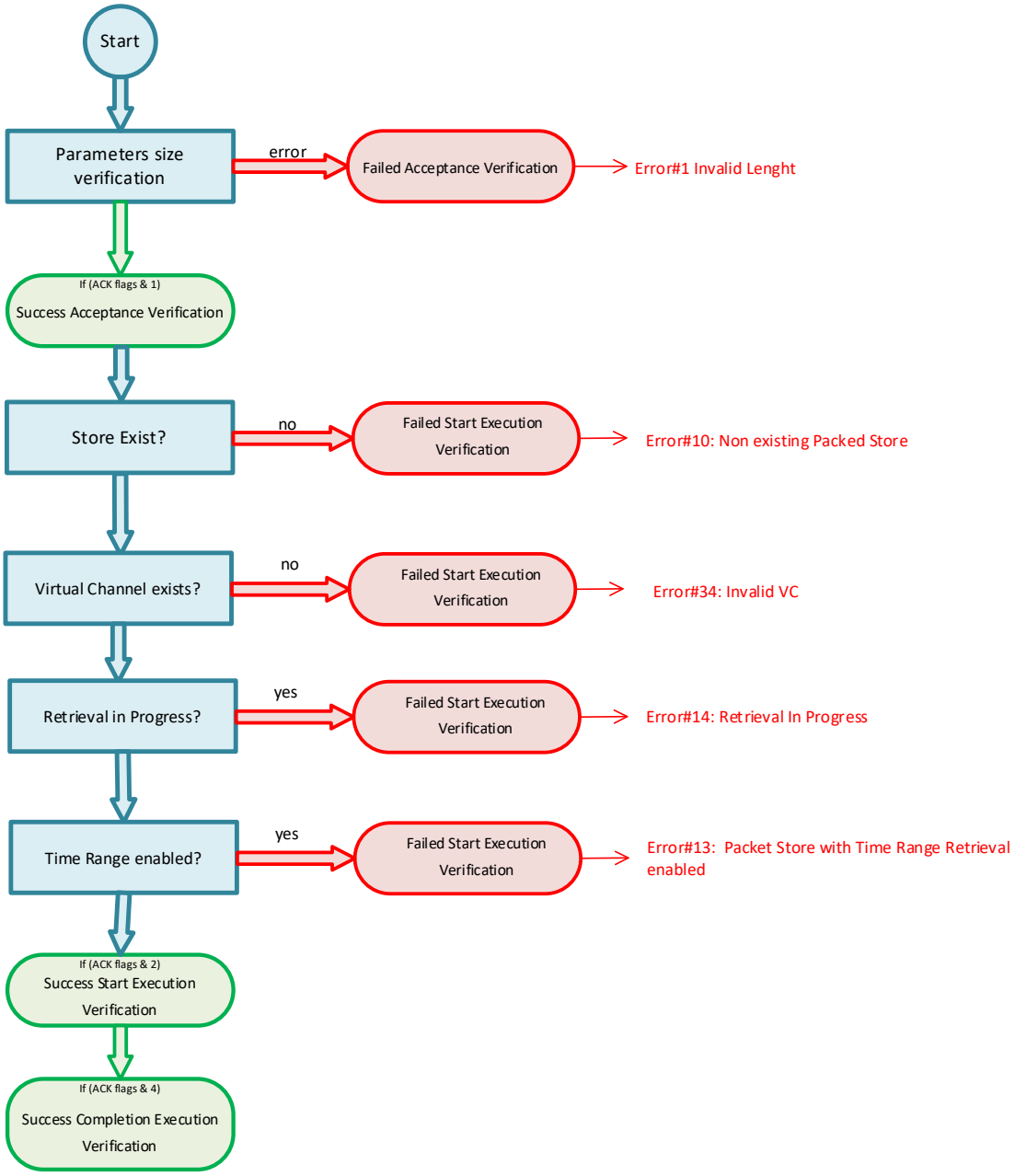
packet store ID	packet store virtual channel
fixed character-string	enumerated

String, Mission Defined Size(**)

uint8

(**) by default size = 15 bytes

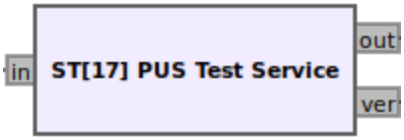
Message request verification flow



2.17 ST[17] TEST

The ***ST[17] Test*** block will received all message request at its input port and if those request are for service ST[17] and for a valid subtype it will check the

request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Messages

In

The message requests input

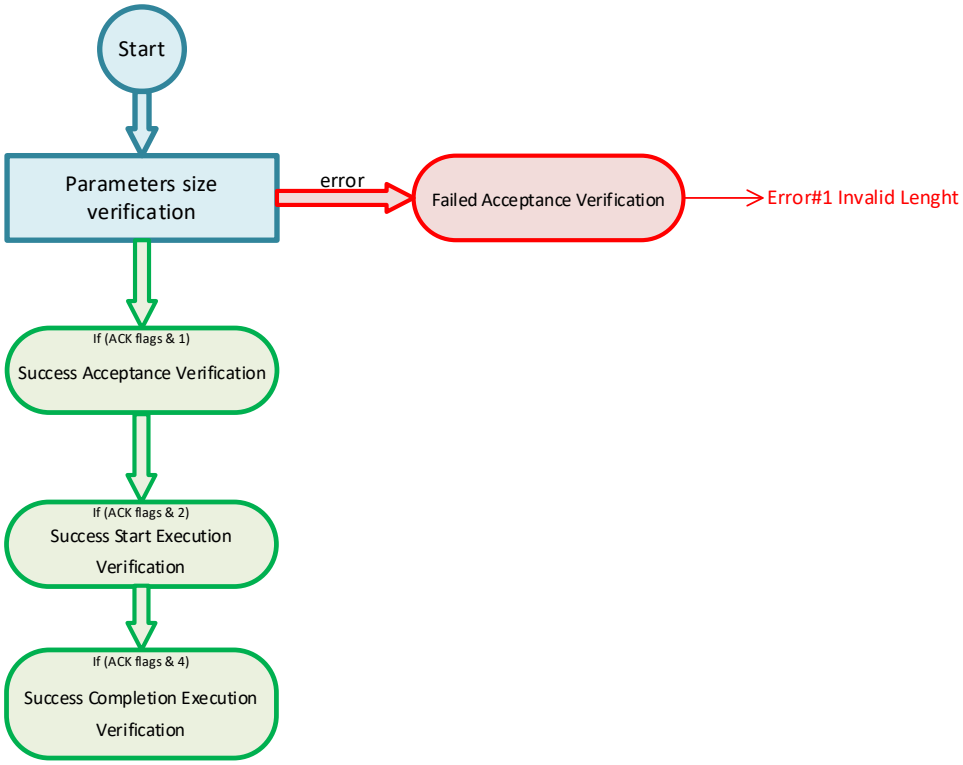
Out

The message report output

Subtypes requests

TC[17,1] perform an are-you-alive connection test

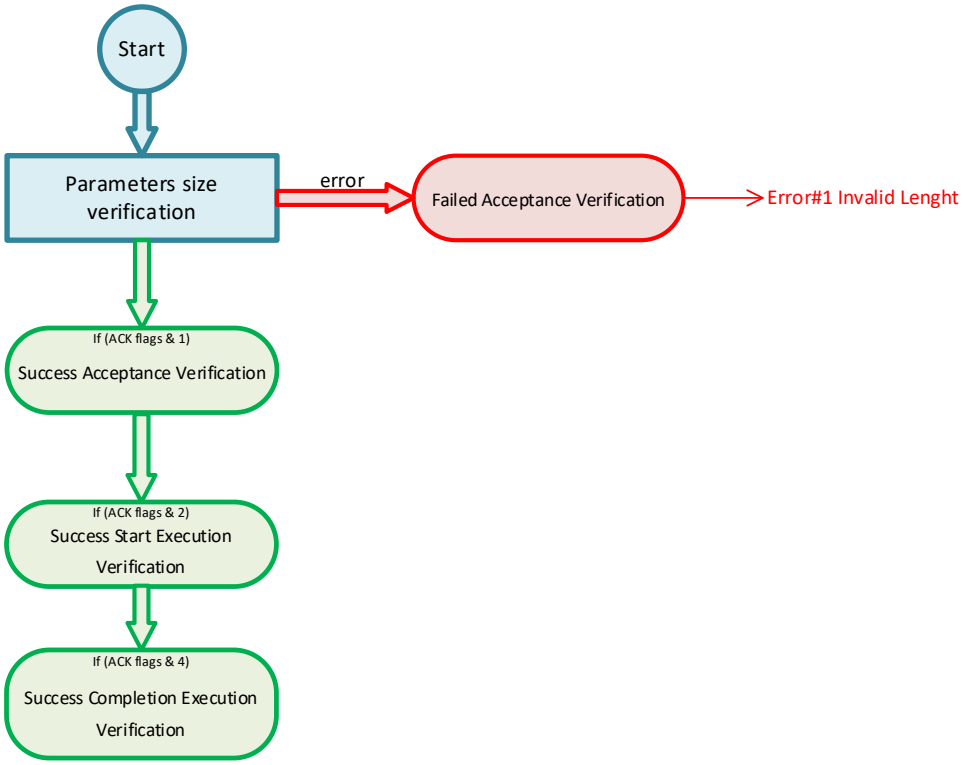
Message request verification flow



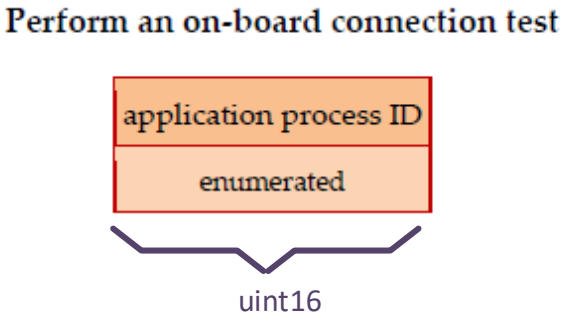
TM[17,2] are-you-alive connection test report

TC[17,3] perform an on-board connection test

Message request verification flow

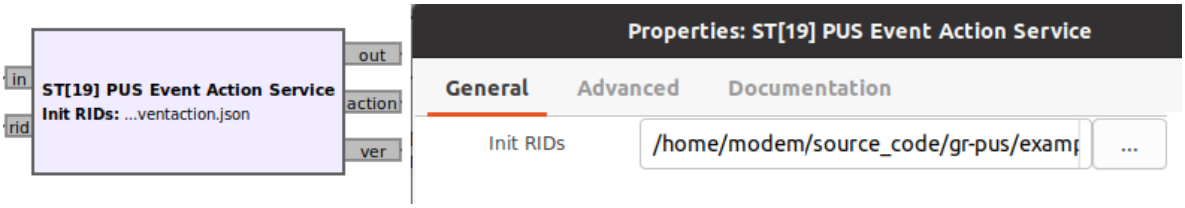


TM[17,4] on-board connection test report



2.18 [ST19]EVENT ACTION

The ***ST[19] Event Action Service*** block will received all message request at its input port and if those request are for service ST[19] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Init RIDs

Path to the json file with the start up RID actions definitions, left empty if no init required

Messages

In

The message requests input

Out

The message report output

rid

The RID message input from On Board Monitoring service

action

The RID action output (message request) triggered by the RID message

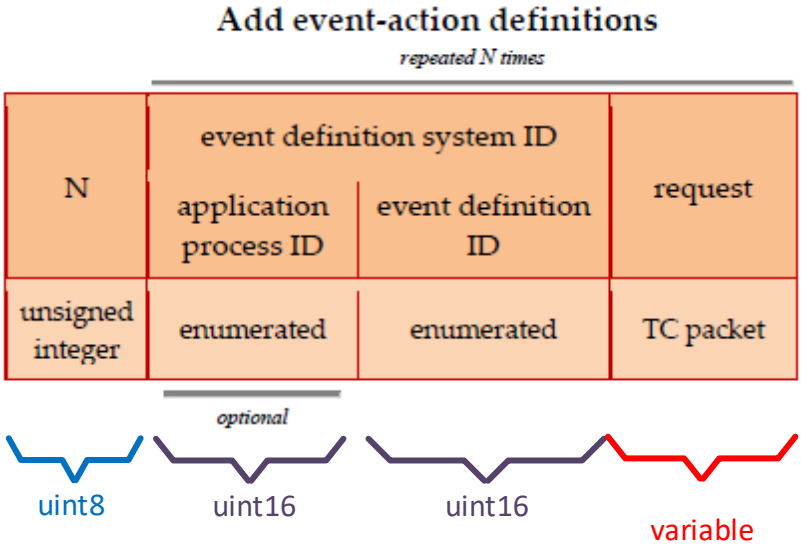
The json init file has next format:

```
{
    "enabled": true,
    "events": [
        {
            "apid": 20,
            "id": 1,
            "enabled": false,
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x09,0x00,0x00,0x02,0x01,0x04"
        },
        {
            "apid": 20,
            "id": 3,
            "enabled": false,
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x19,0x00,0x00,0x00,0x00,0x00"
        },
        {
            "apid": 20,
            "id": 6,
            "enabled": false,
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x1f,0x00,0x00,0x02,0x01,0x04"
        },
```

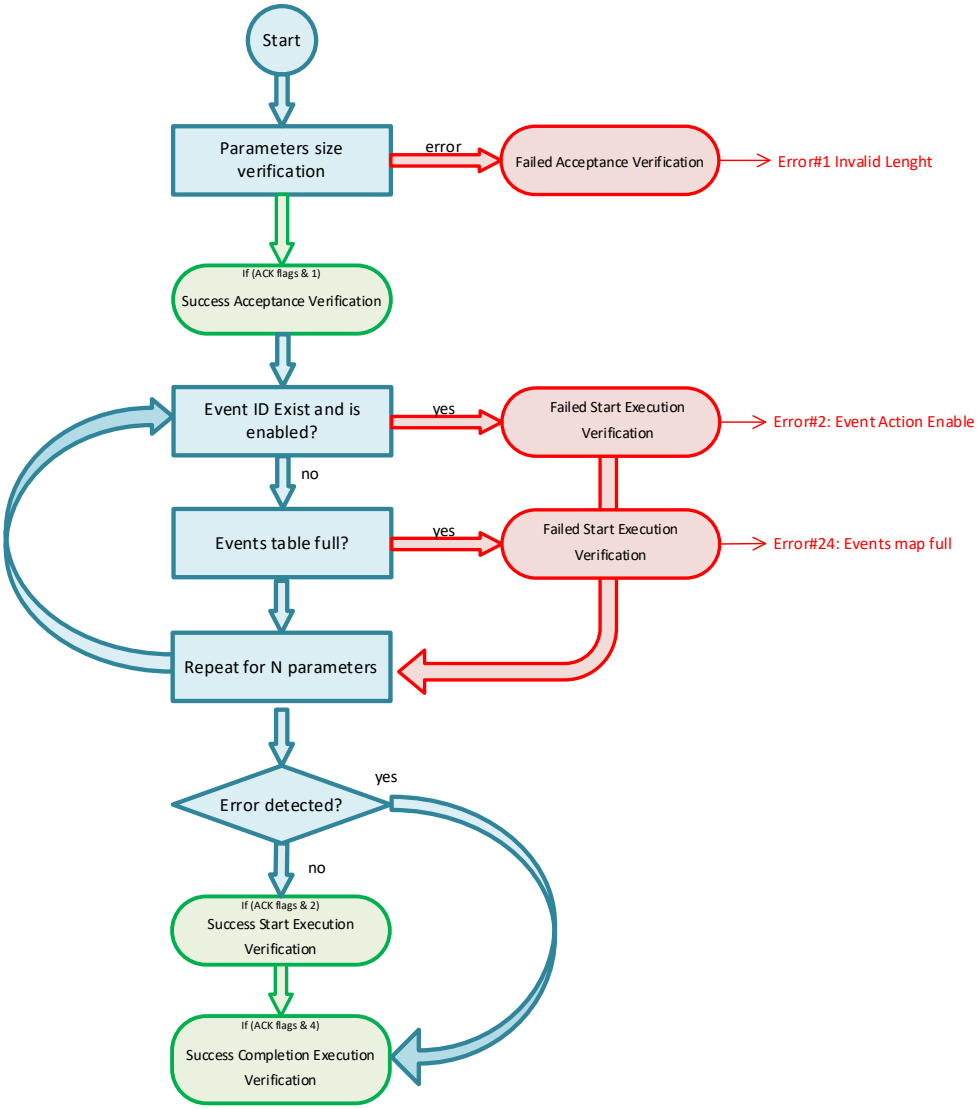
Data is the message request, as hex byte string, to be sent when a RID matching ID is received in the rid input port.

Subtypes requests

TC[19,1] add event-action definitions

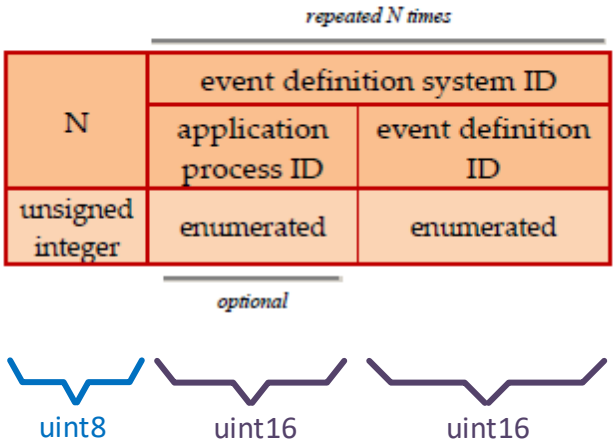


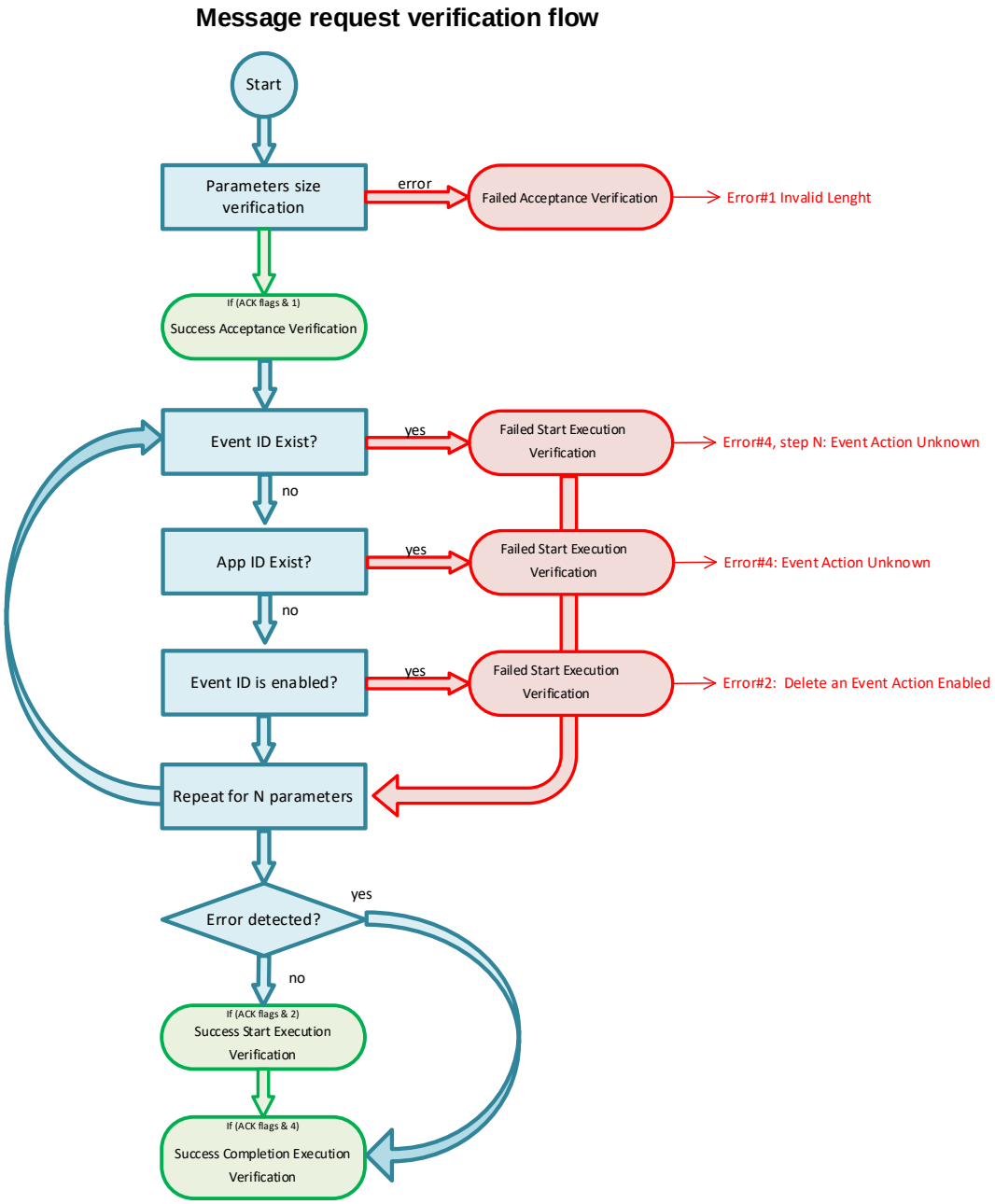
Message request verification flow



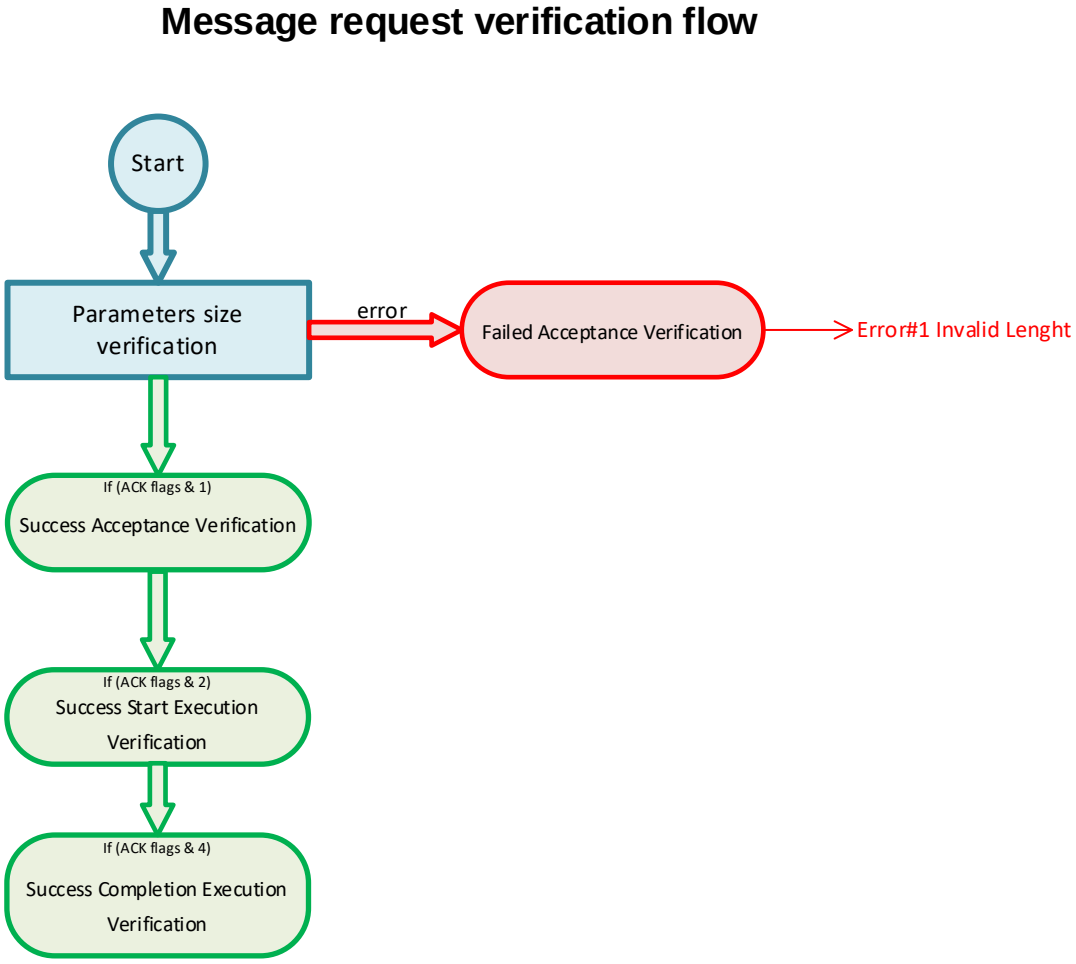
TC[19,2] delete event-action definitions

Delete event-action definitions

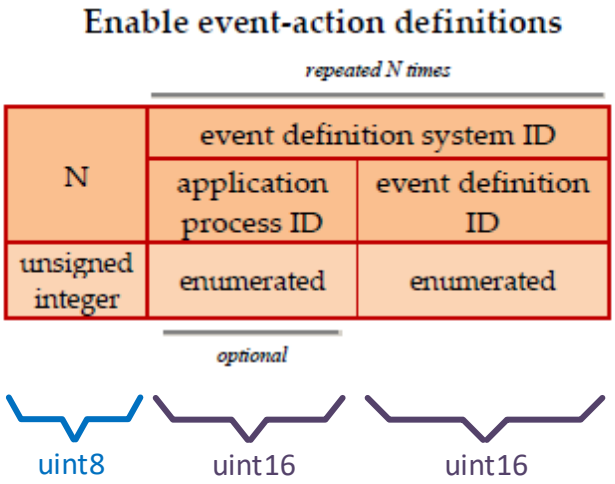




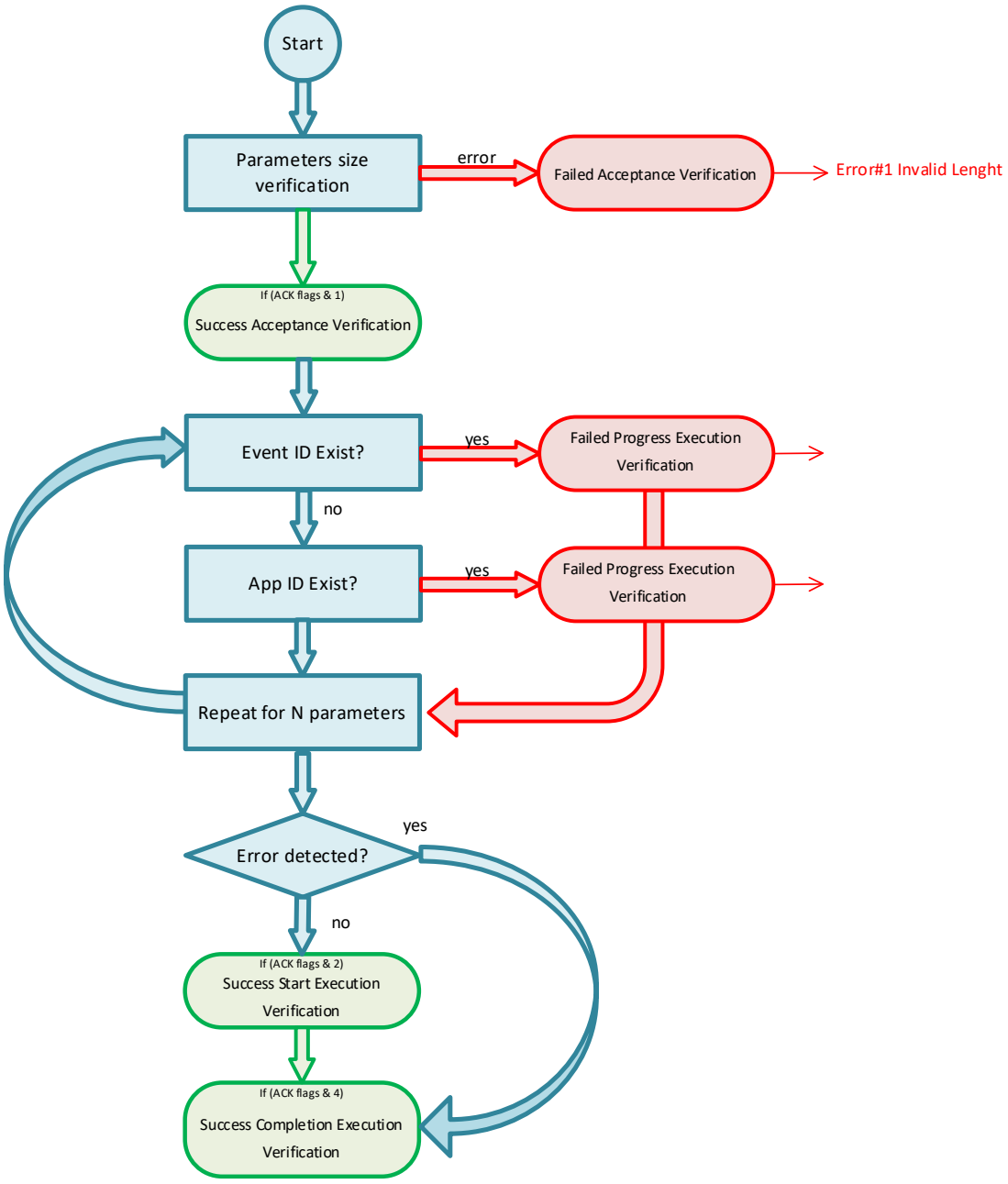
TC[19,3] delete all event-action definitions



TC[19,4] enable event-action definitions



Message request verification flow



TC[19,5] disable event-action definitions

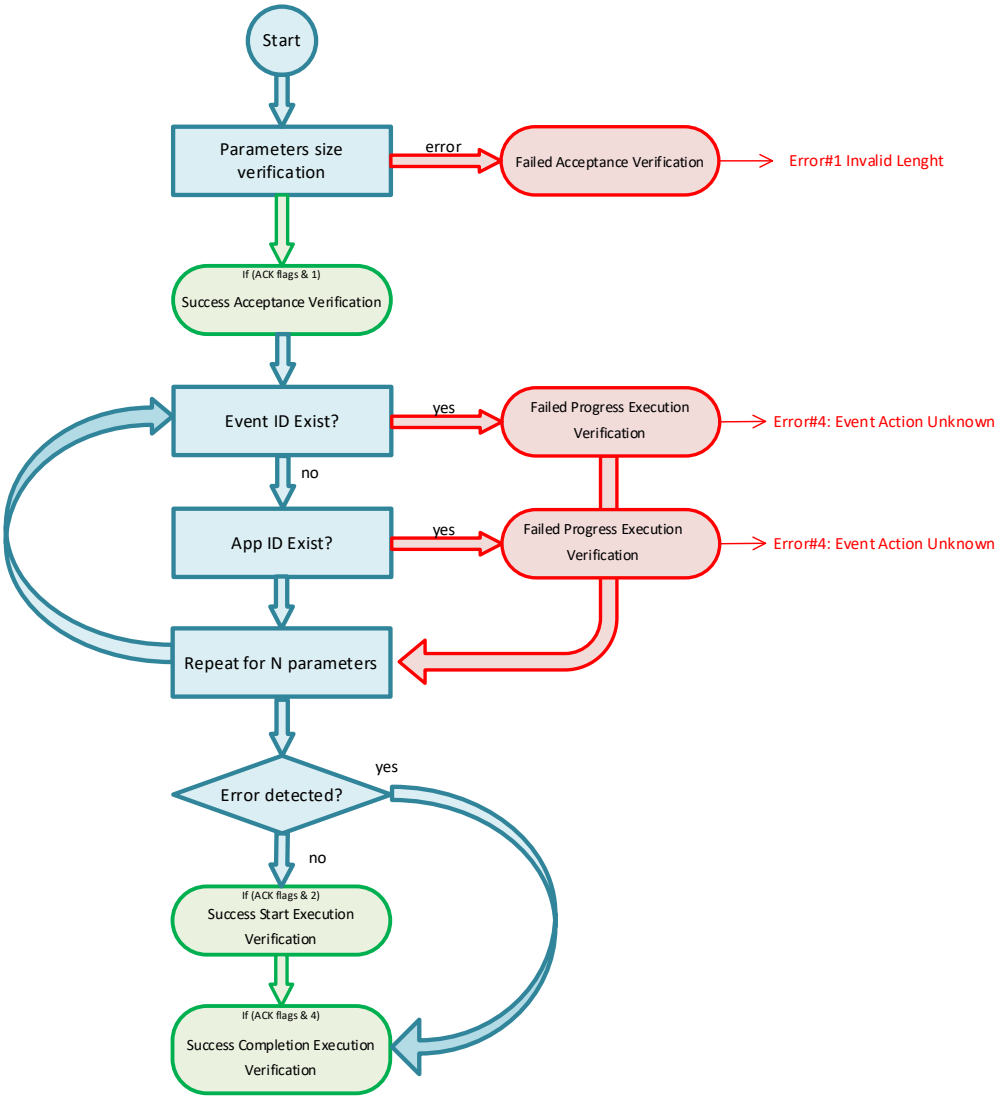
Disable event-action definitions

repeated N times

N	event definition system ID	
	application process ID	event definition ID
unsigned integer	enumerated	enumerated

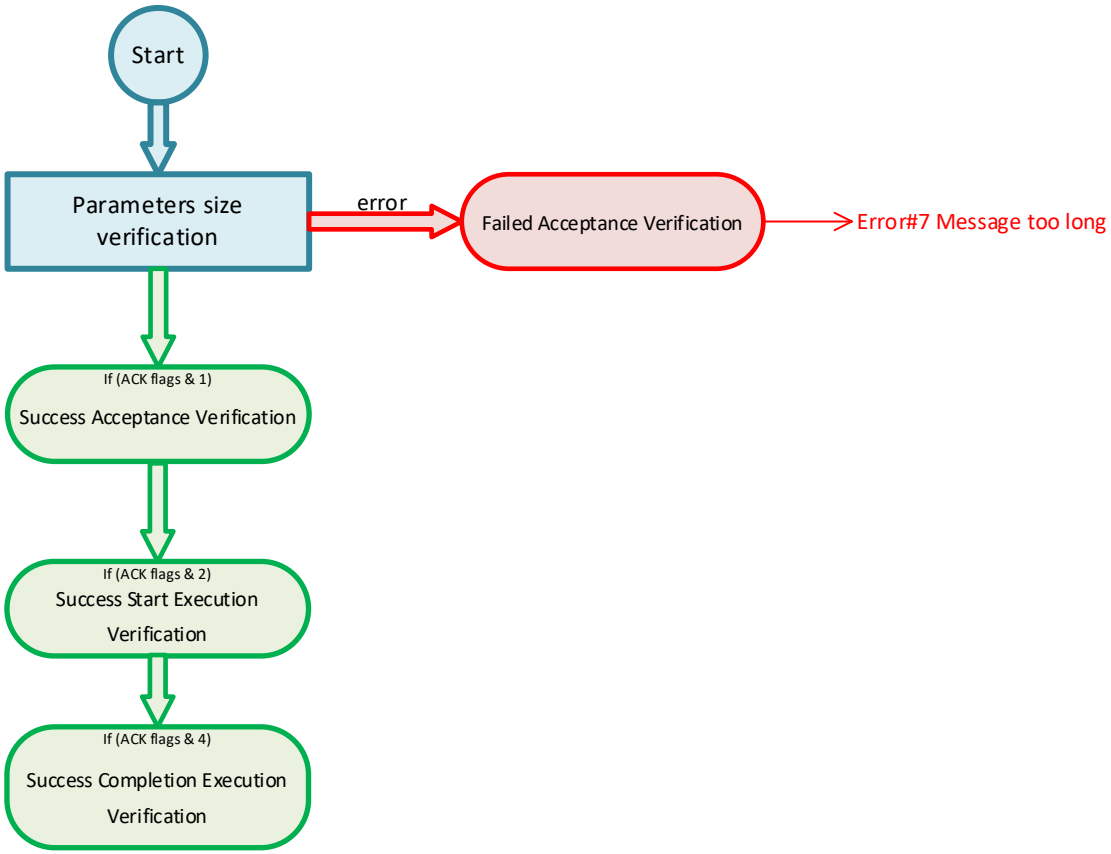
optional

Message request verification flow

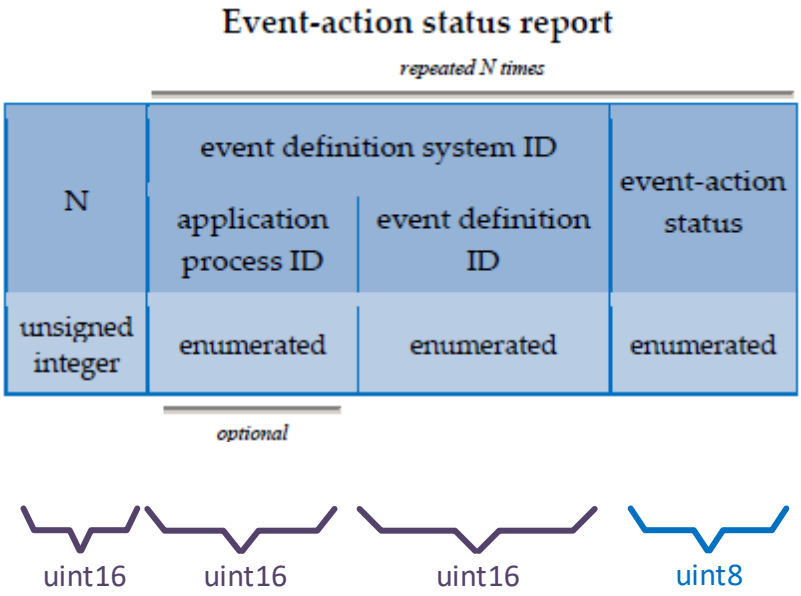


TC[19,6] report the status of each event-action definition

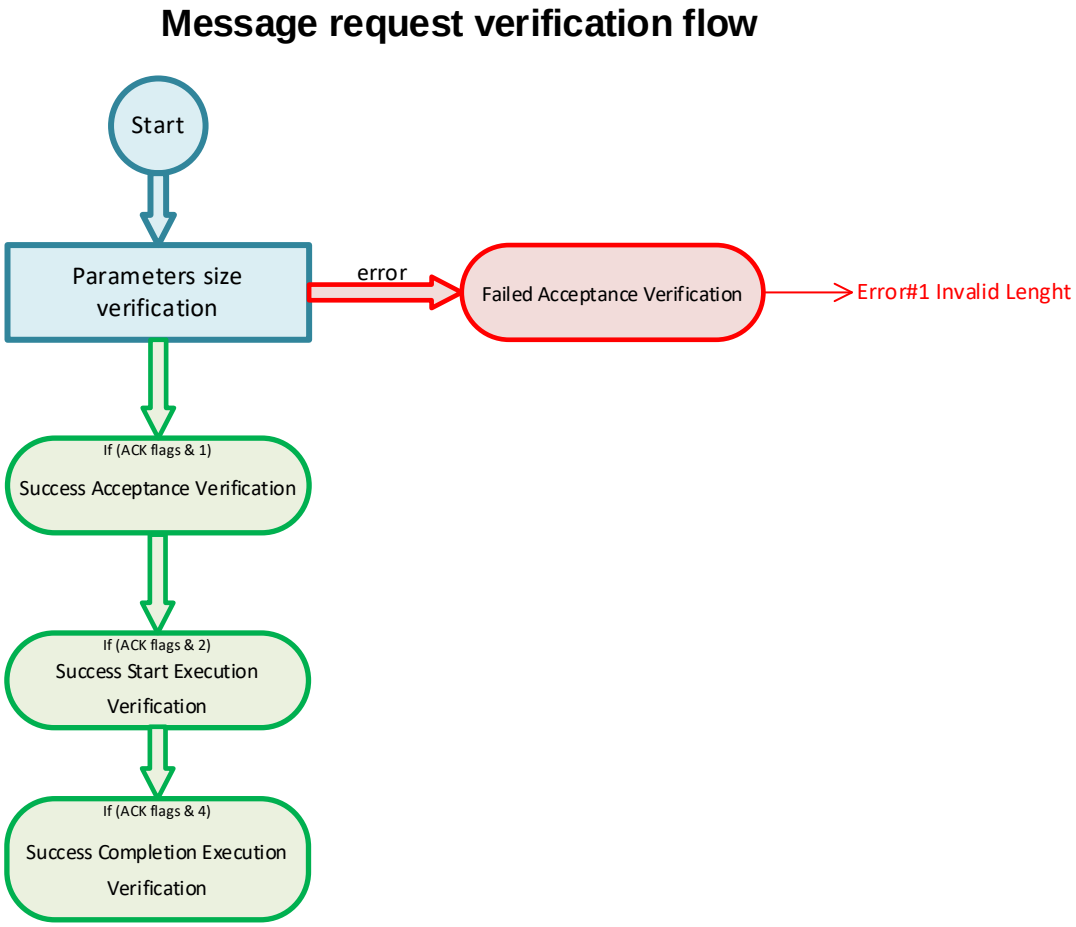
Message request verification flow



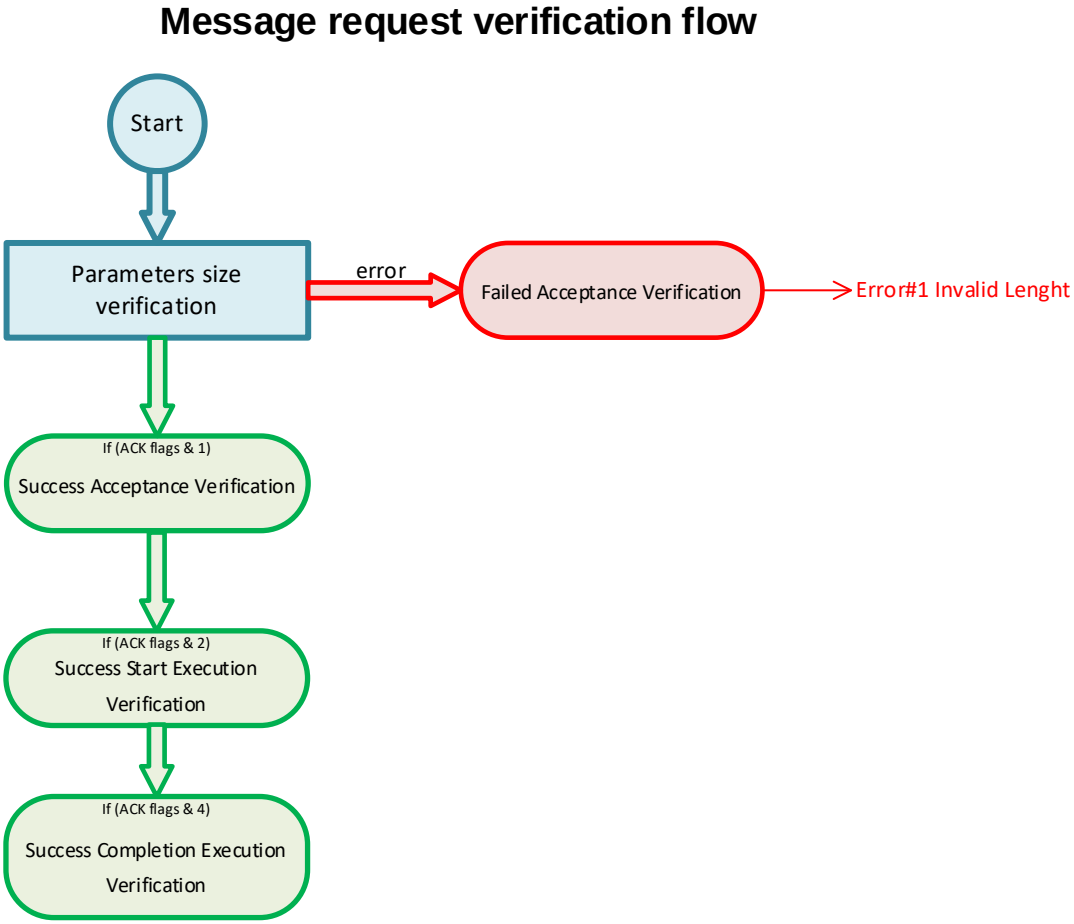
TM[19,7] event-action status report



TC[19,8] enable the event-action function



TC[19,9] disable the event-action function



TC[19,10] report event-action definitions

Report event-action definitions

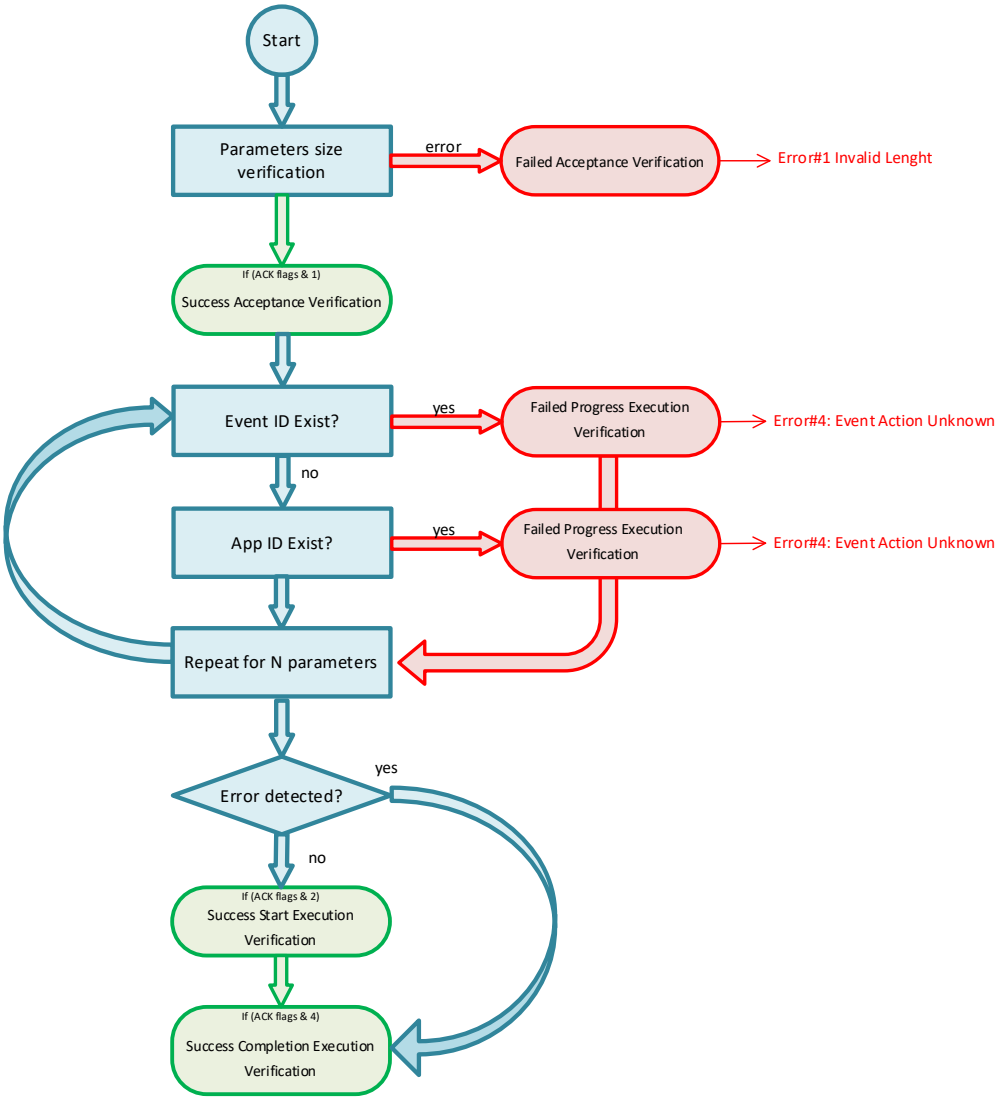
repeated N times

N	event definition system ID	
	application process ID	event definition ID
unsigned integer	enumerated	enumerated

optional

uint8 uint16 uint16

Message request verification flow



TM[19,11] event-action definition report

Event-action definition report

repeated N times

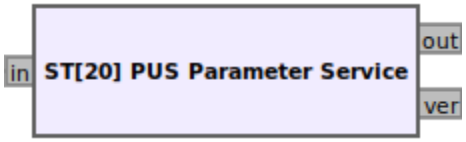
N	event definition system ID		event-action status	request
	application process ID	event definition ID		
unsigned integer	enumerated	enumerated	enumerated	TC packet

optional

uint8 uint16 uint16 uint8

2.19 ST[20] PARAMETER MANAGEMENT

The ***ST[20] Parameter Management*** block will received all message request at its input port and if those request are for service ST[20] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Messages

In

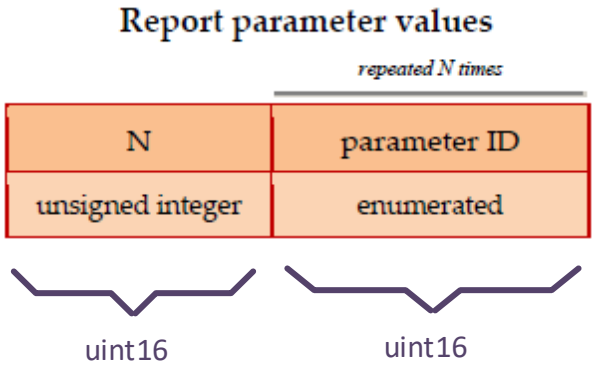
The message requests input

Out

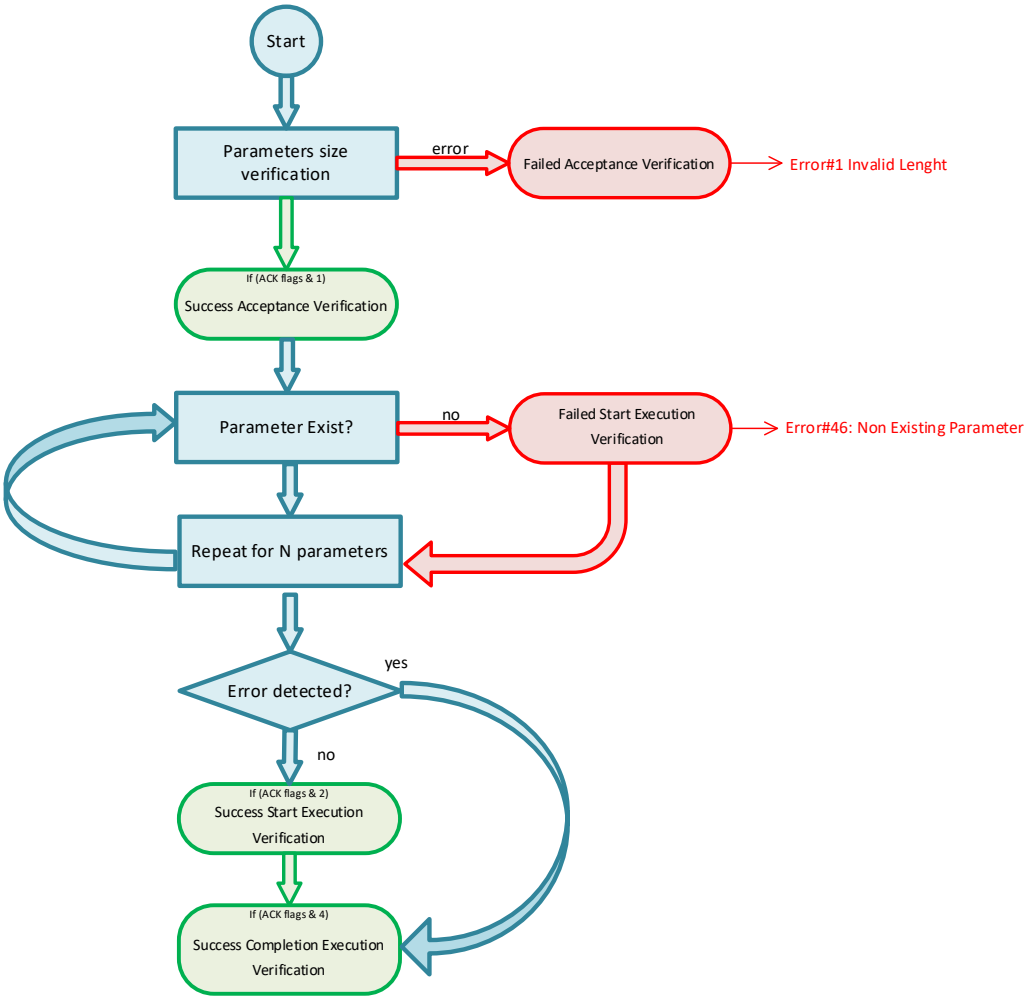
The message report output

Subtypes requests

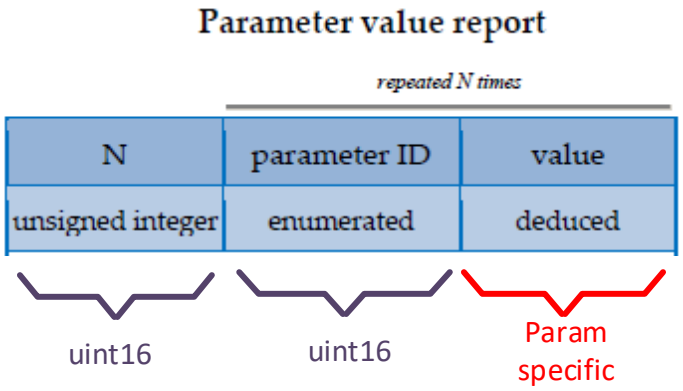
TC[20,1] report parameter values



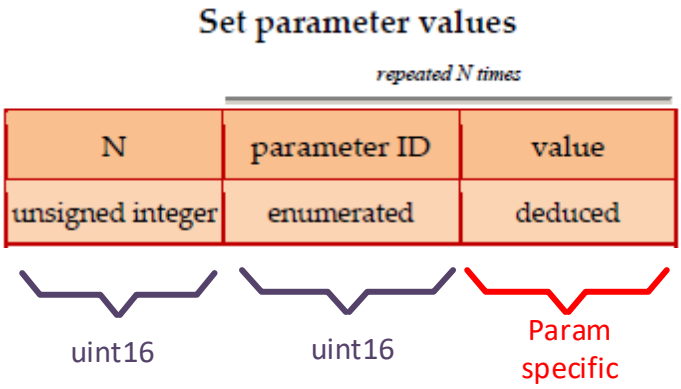
Message request verification flow



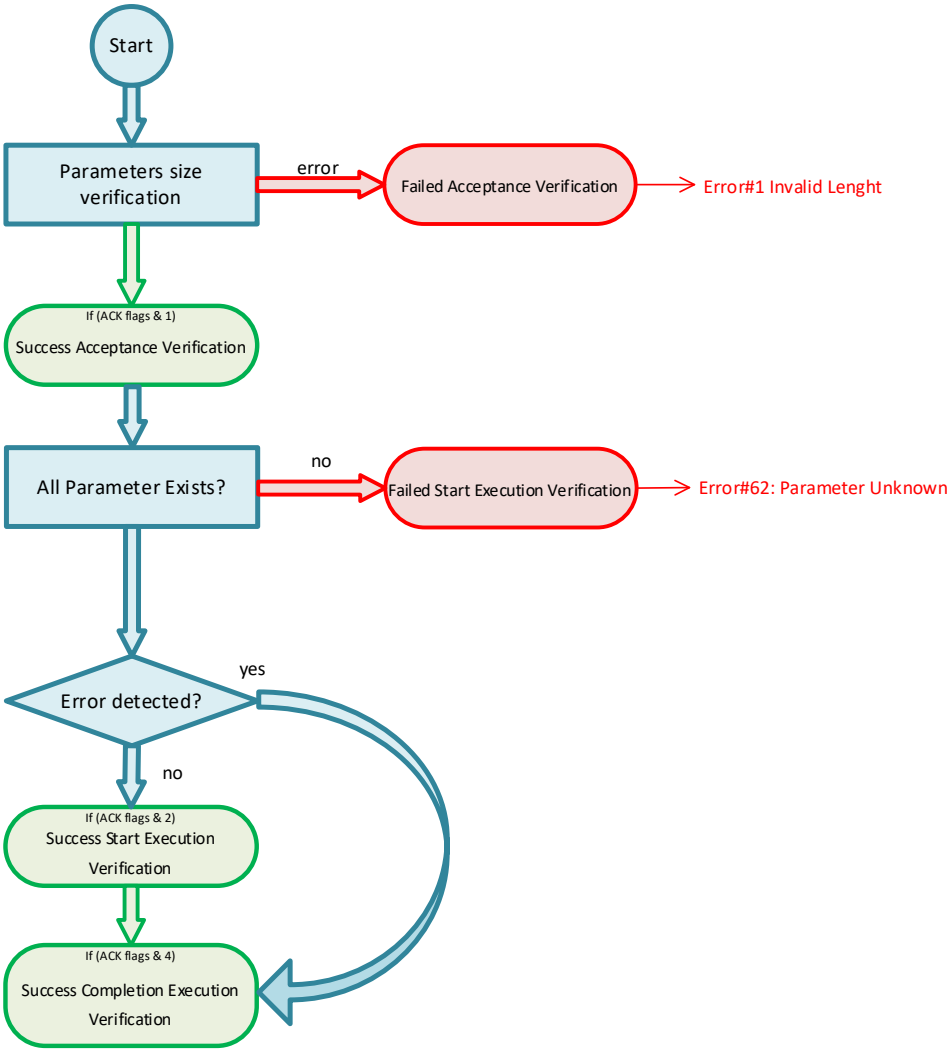
TM[20,2] parameter value report



TC[20,3] set parameter values



Message request verification flow



2.20 ST[21] REQUEST SEQUENCING

The ***ST[21] Request Sequencing Service*** block will received all message request at its input port and if those request are for service ST[21] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected

Parameters

(*R*): *Run-time adjustable*

Init RIDs

Path to the json file with the start up RID actions definitions, left empty if no init required

Messages

In

The message requests input

Out

The message report output

release

The message requests of the action sequences RID will be send thru this output port

The json init file has next format:

```
{
    "sequenceStores": [
        {
                "name": "TestSeq#1",
                "seqFile": "../../../examples/init_reqseq1.json"
        },
        {
                "name": "TestSeq#2",
                "seqFile": "../../../examples/init_reqseq2.json"
        },
        {
                "name": "TestSeq#3",
                "seqFile": "../../../examples/init_reqseq3.json"
        }
    ]
}
```

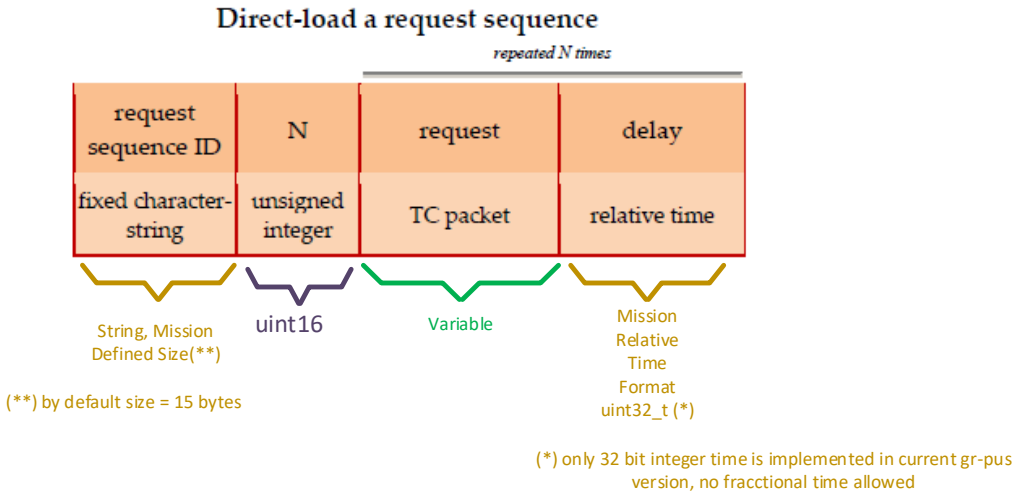
Each seqFile has next format:

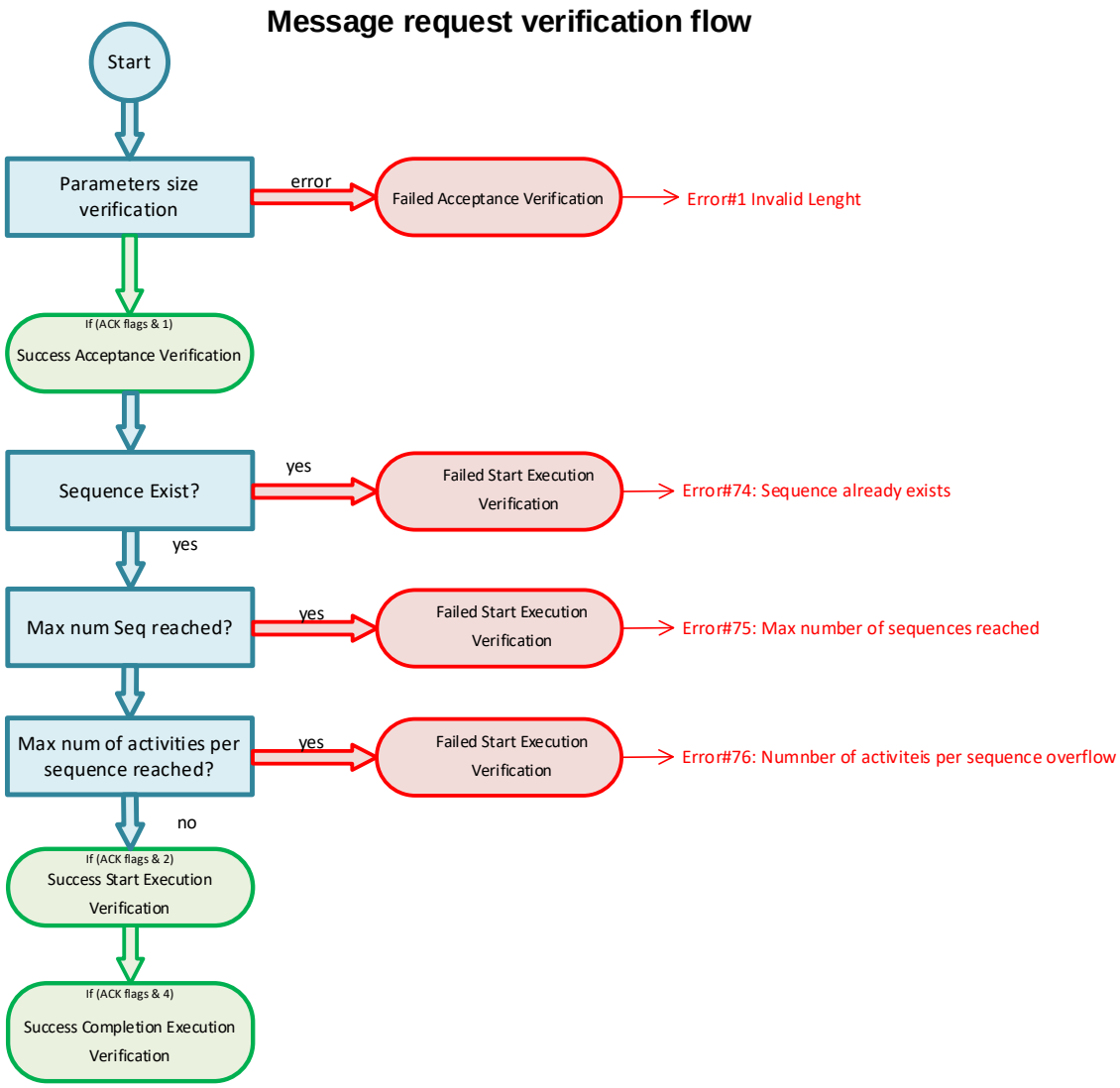
```
{
    "sequence": [
        {
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x09,0x00,0x00,0x02,0x01,0x04",
            "delay": 2
        },
        {
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x19,0x00,0x00,0x00,0x00,0x00",
            "delay": 0
        },
        {
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x1f,0x00,0x00,0x02,0x01,0x04",
            "delay": 0
        },
        {
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x1b,0x00,0x00,0x02,0x01,0x04",
            "delay": 5
        },
        {
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x1b,0x00,0x00,0x02,0x01,0x04",
            "delay": 10
        },
        {
            "data": "0x18,0x17,0xc0,0x00,0x00,0x07,0x20,0x03,0x1b,0x00,0x00,0x02,0x01,0x04",
            "delay": 0
        }
    ]
}
```

The delay is in sec, if several request shall be released at same time set delay = 0, they will released in the proper order but with no waiting time.
Data is the message request to be released in each step in hex format

Subtypes requests

TC[21,1] direct-load a request sequence





TC[21,2] load a request sequence by reference

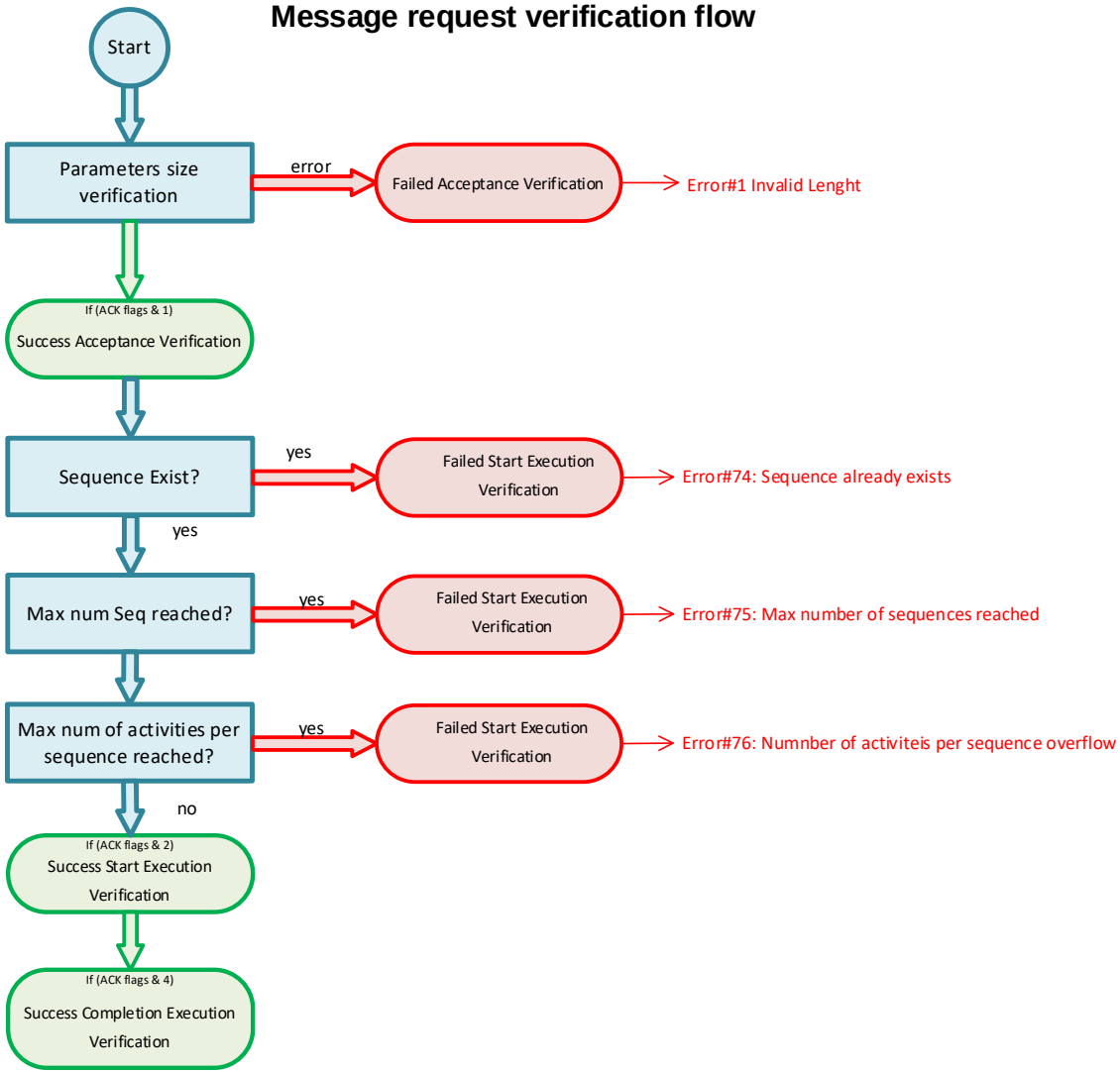
Load a request sequence by reference

request sequence ID	file path	
	repository path	file name
fixed character-string	variable character-string	variable character-string
	optional	

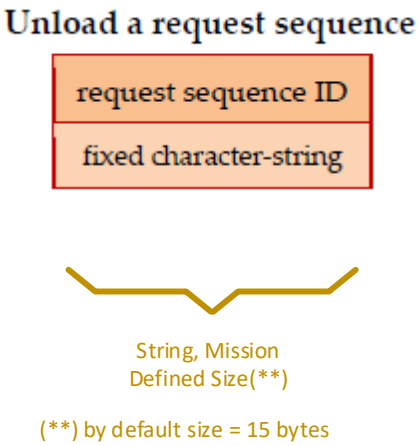
String, Mission Defined Size(**) | String, variable | String, variable

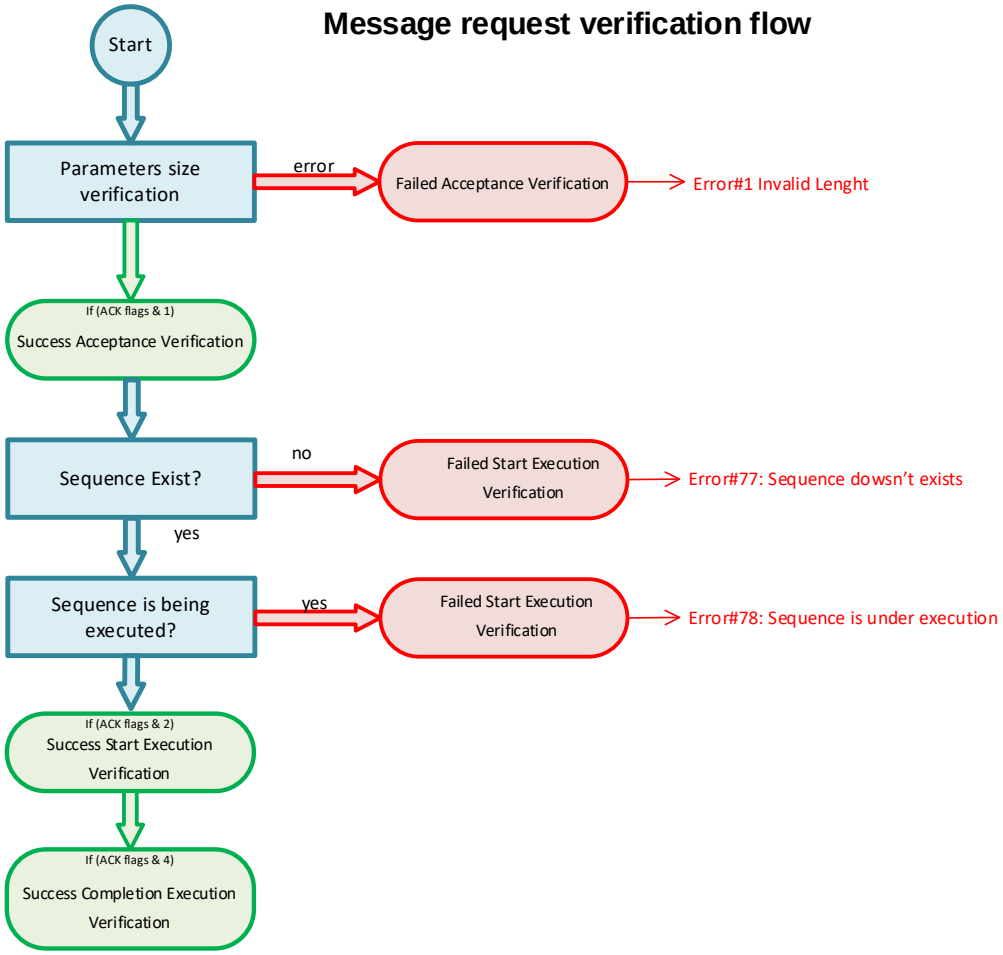
(**) by default size = 15 bytes

Message request verification flow

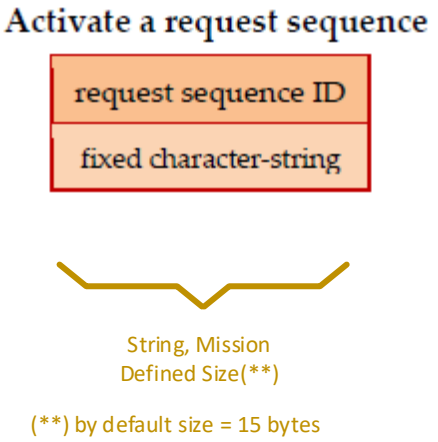


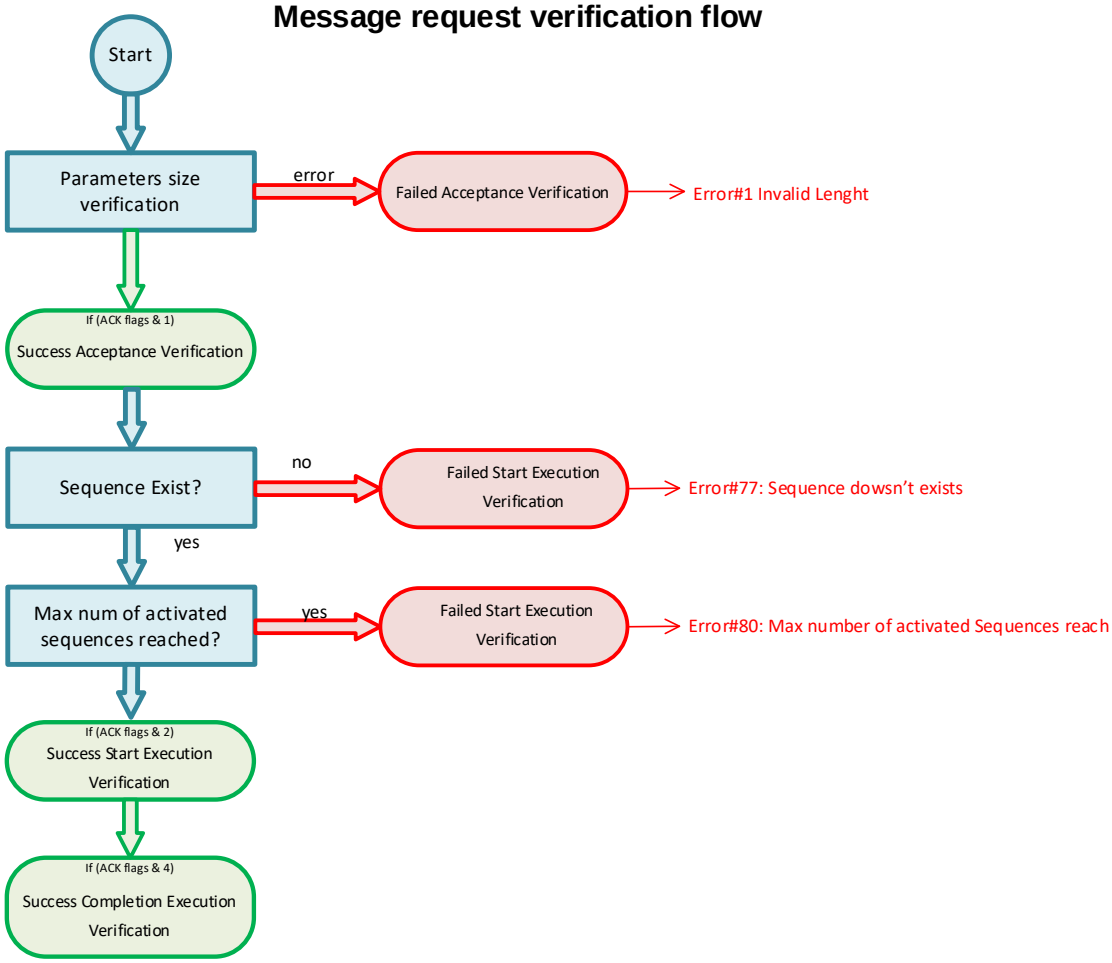
TC[21,3] unload a request sequence



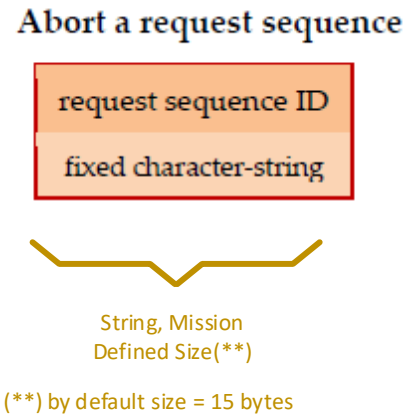


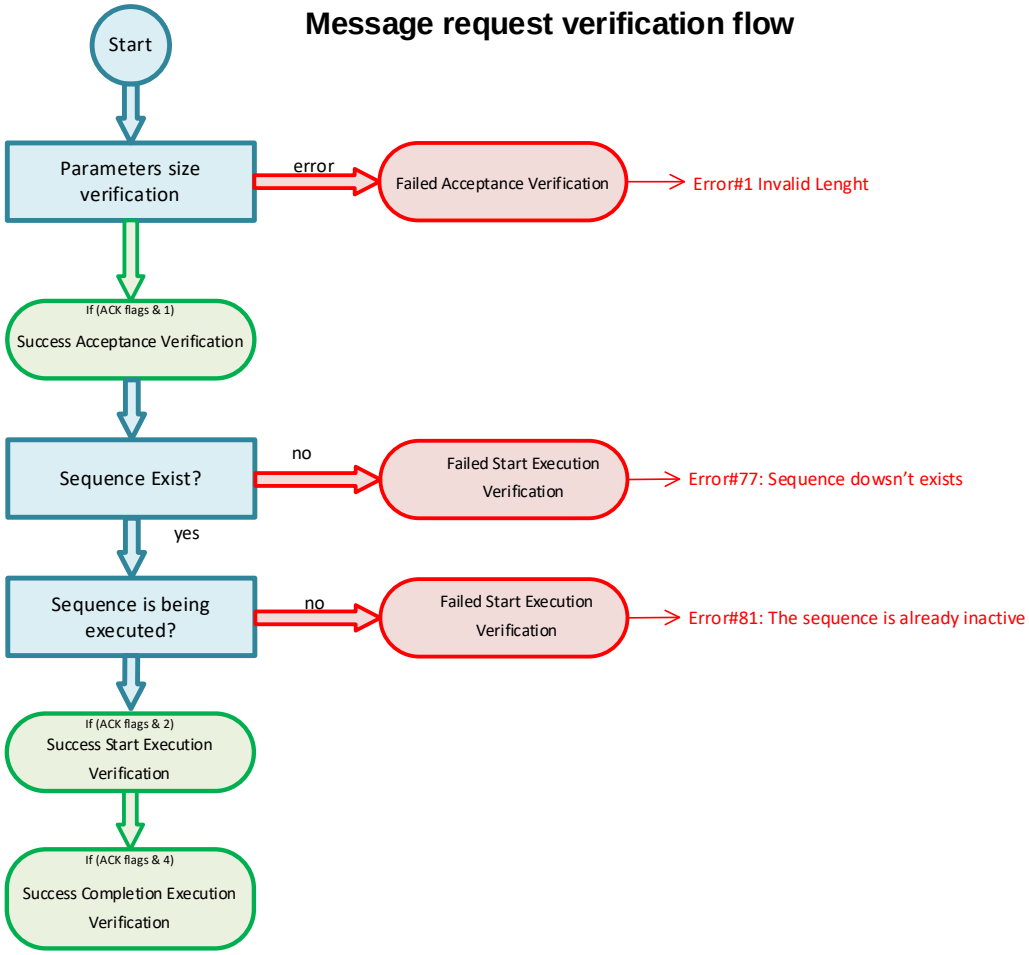
TC[21,4] activate a request sequence





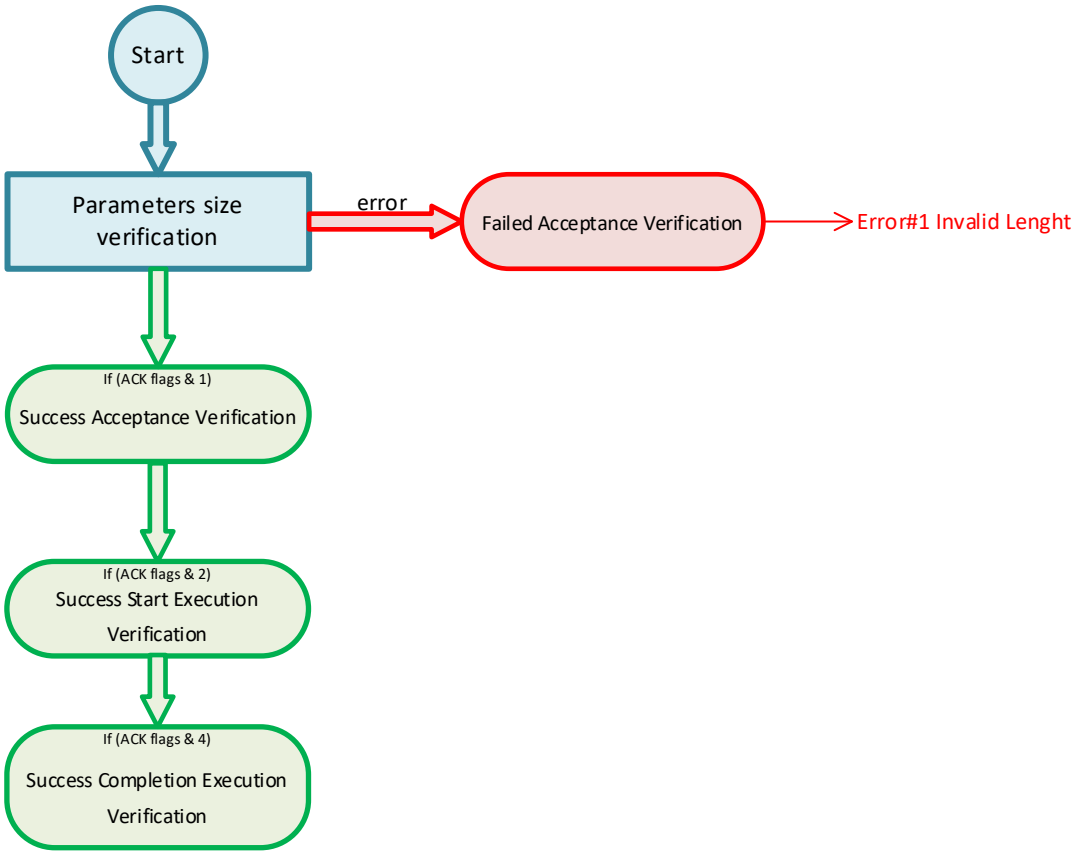
TC[21,5] abort a request sequence





TC[21,6] report the execution status of each request sequence

Message request verification flow



TM[21,7] request sequence execution status report

Request sequence execution status report

repeated N times

N	request sequence ID	execution status
unsigned integer	fixed character-string	enumerated

uint16 — String, Mission Defined Size(**) — uint8

(**) by default size = 15 bytes

TC[21,8] load by reference and activate a request sequence

Load by reference and activate a request sequence

request sequence ID	file path	
	repository path	file name
fixed character-string	variable character-string	variable character-string

optional

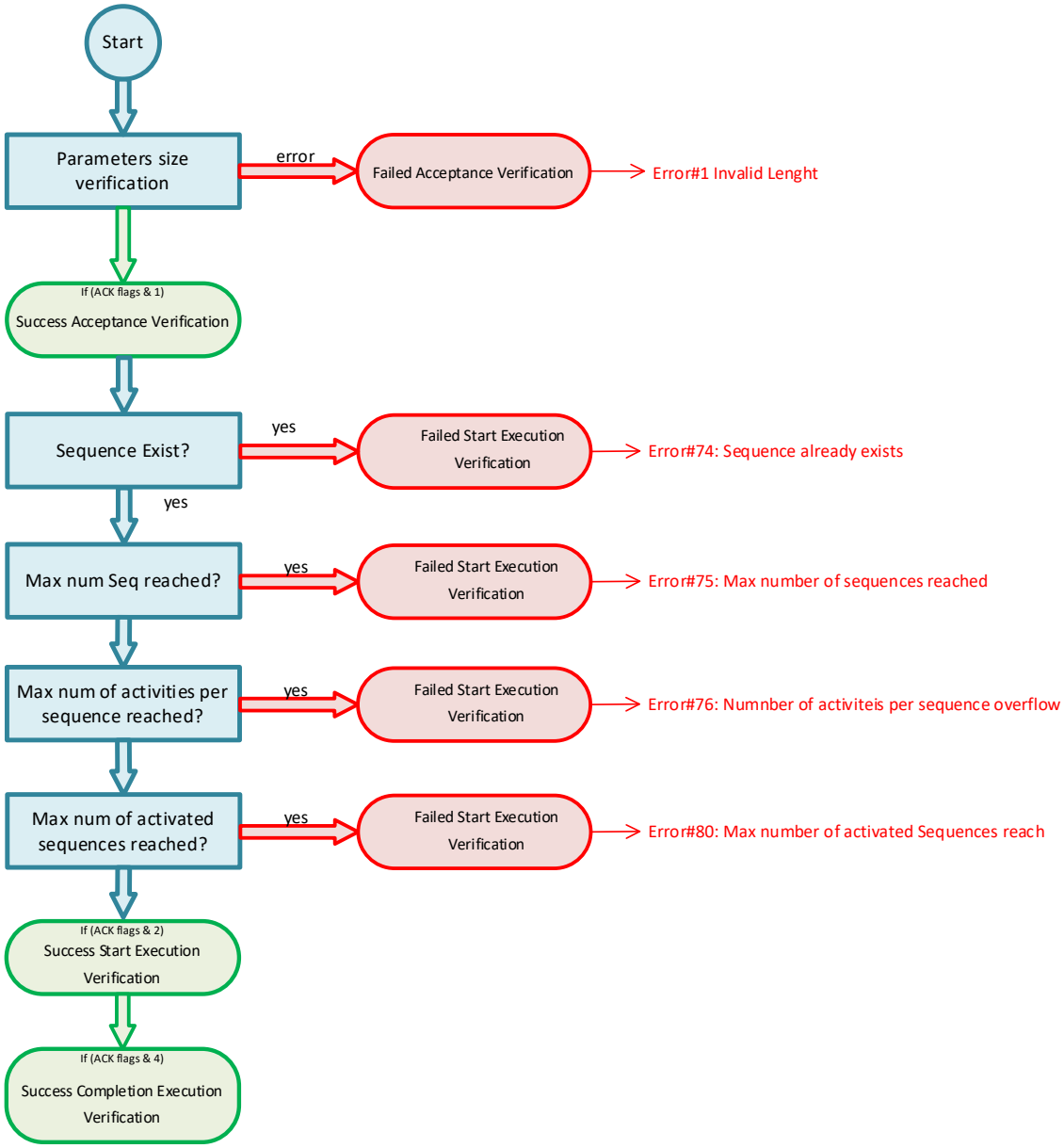
String, Mission Defined Size(**)

String, variable

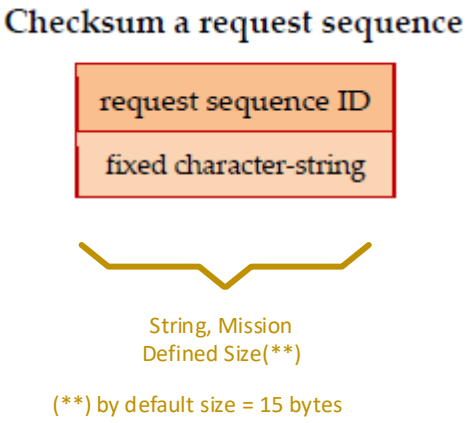
String, variable

(**) by default size = 15 bytes

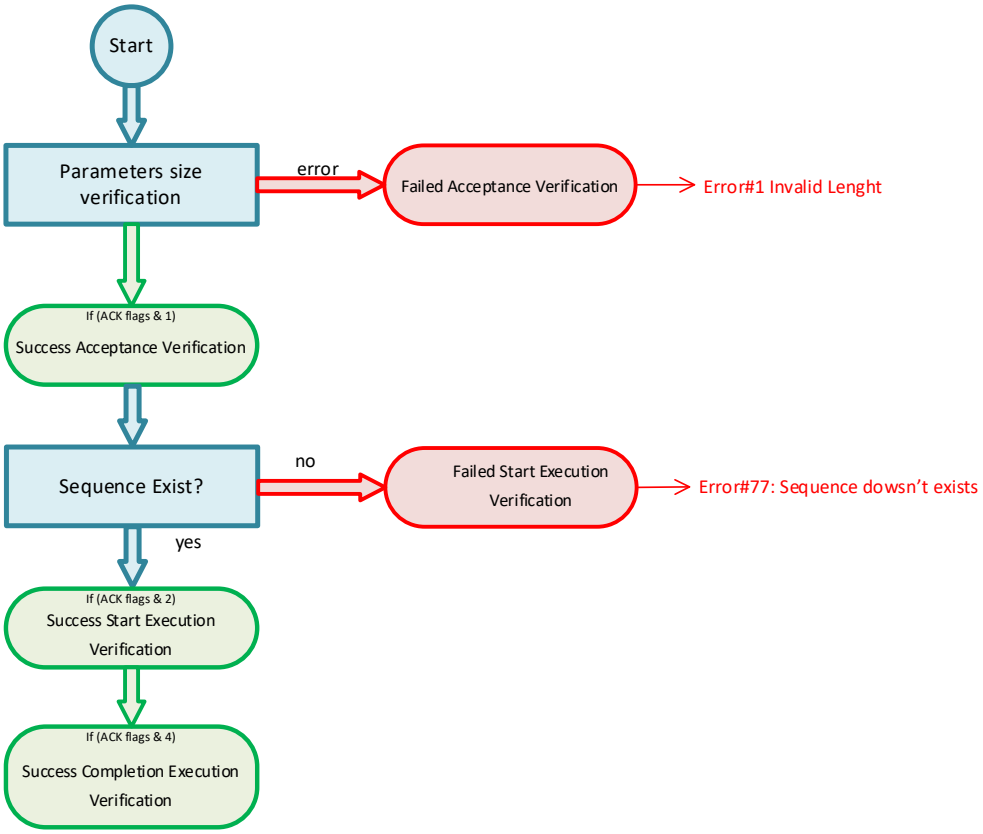
Message request verification flow



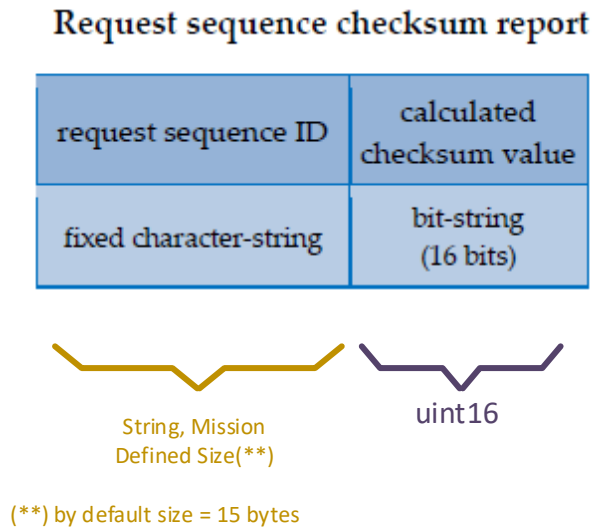
TC[21,9] checksum a request sequence



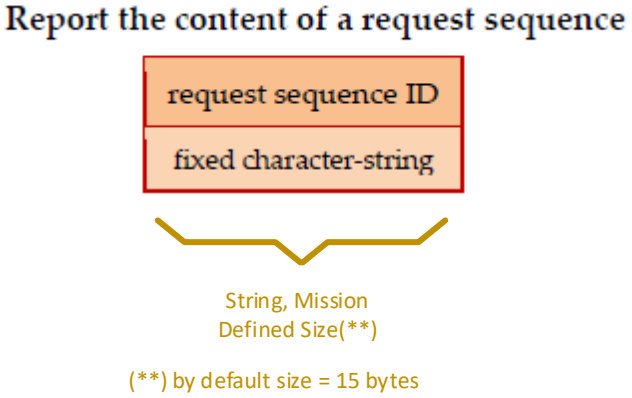
Message request verification flow



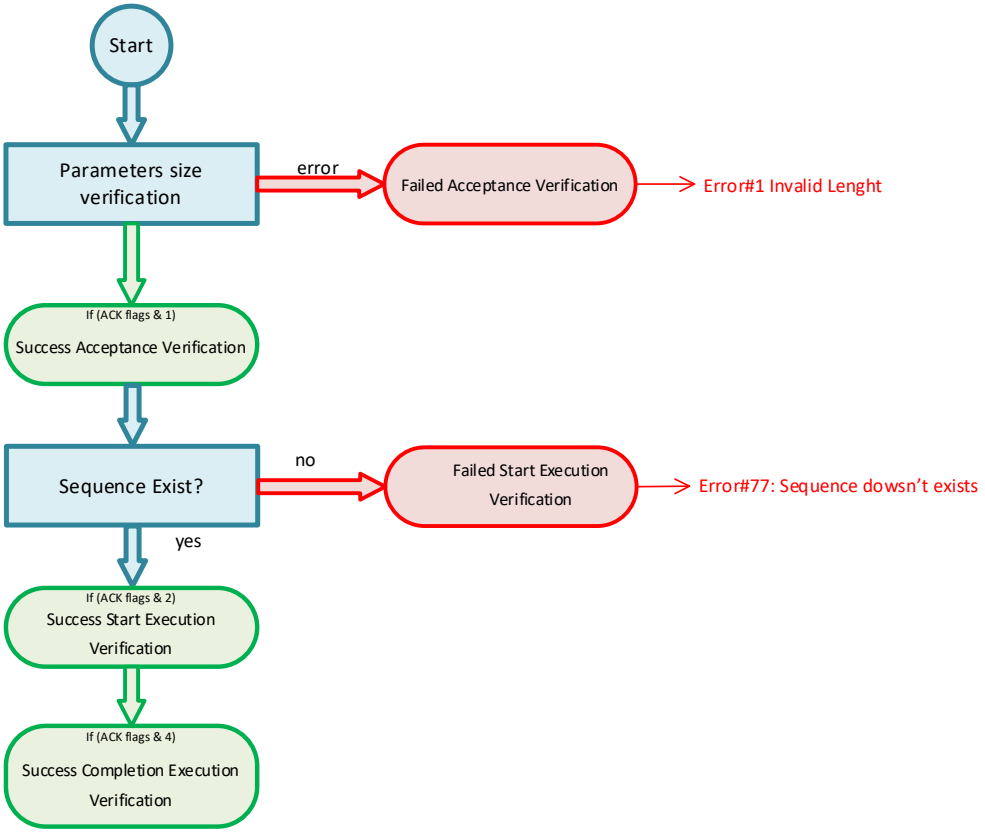
TM[21,10] request sequence checksum report



TC[21,11] report the content of a request sequence



Message request verification flow



TM[21,12] request sequence content report

Request sequence content report

repeated N times

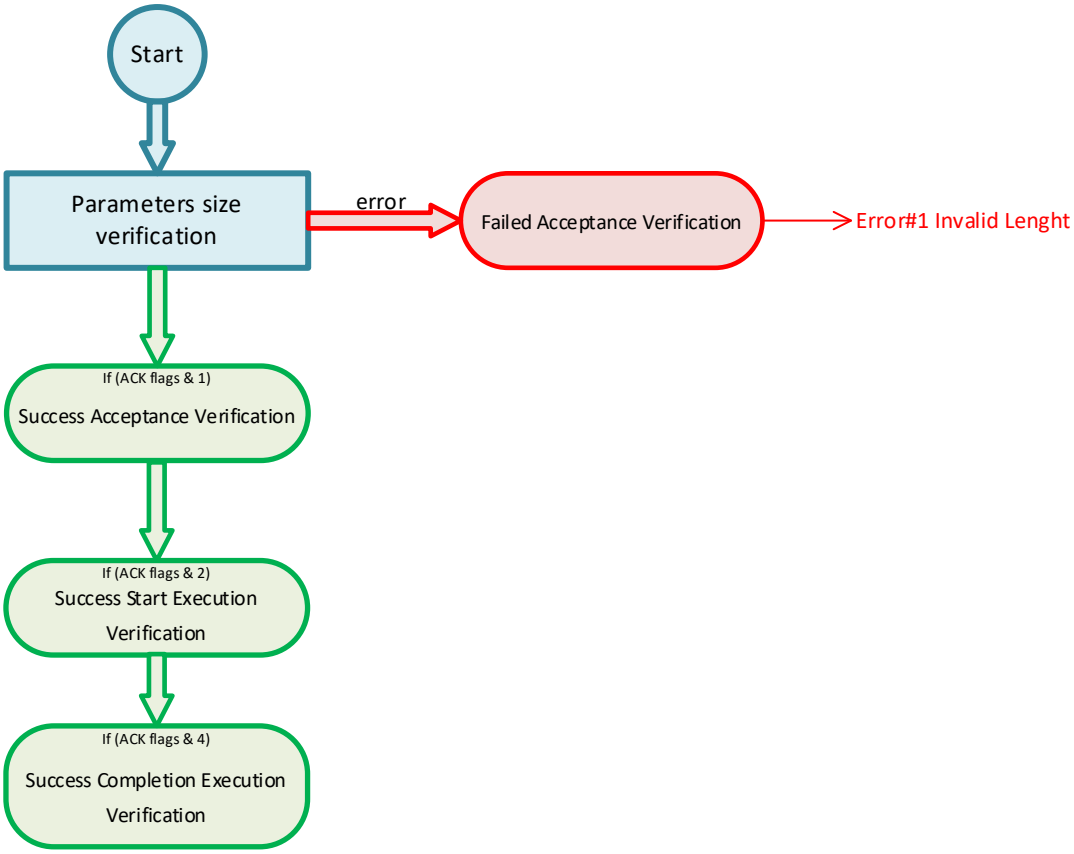
request sequence ID	N	request	delay
fixed character-string	unsigned integer	TC packet	relative time
String, Mission Defined Size(**)	uint16	Variable	Mission Relative Time Format uint32_t (*)

(**) by default size = 15 bytes

(*) only 32 bit integer time is implemented in current gr-pus version, no fracctional time allowed

TC[21,13] abort all request sequences and report

Message request verification flow



TM[21,14] aborted request sequence report

Aborted request sequence report

repeated N times

N	request sequence ID
unsigned integer	fixed character-string

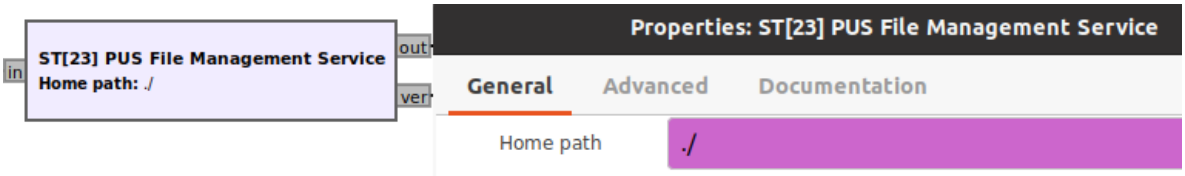
uint16

String, Mission Defined Size(**)

(**) by default size = 15 bytes

2.21 ST[23] FILE MANAGEMENT

The ***ST[23] File Management Service*** block will received all message request at its input port and if those request are for service ST[23] and for a valid subtype it will check the request fields size and then executed the request, otherwise the request will be rejected



Parameters

(*R*): *Run-time adjustable*

Home path

The home path to be added at the beginning of all message request paths as home path, left empty if you are planning to use full absolute paths in the requests

Messages

In

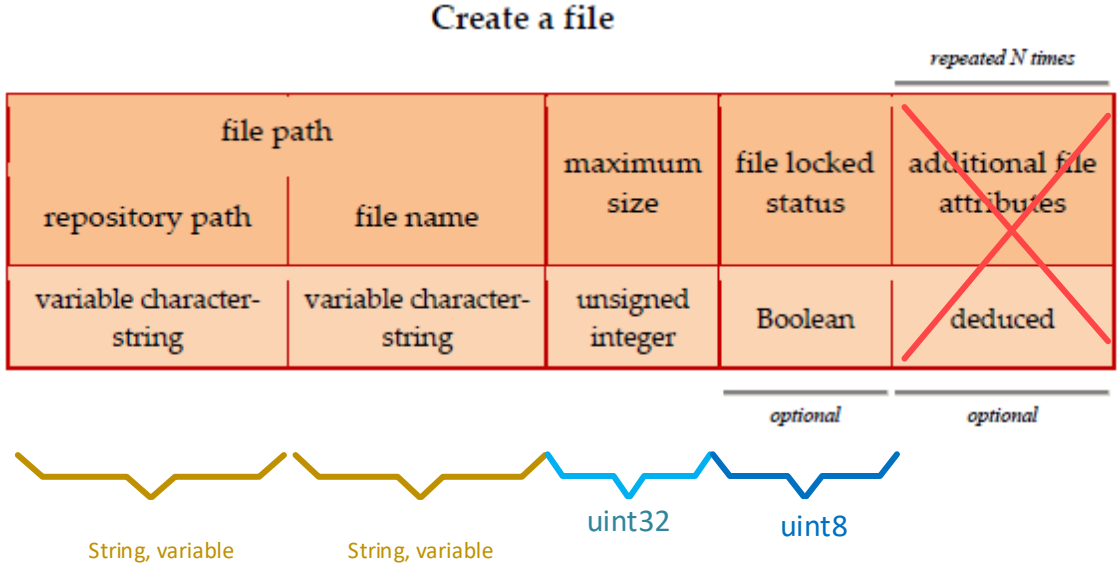
The message requests input

Out

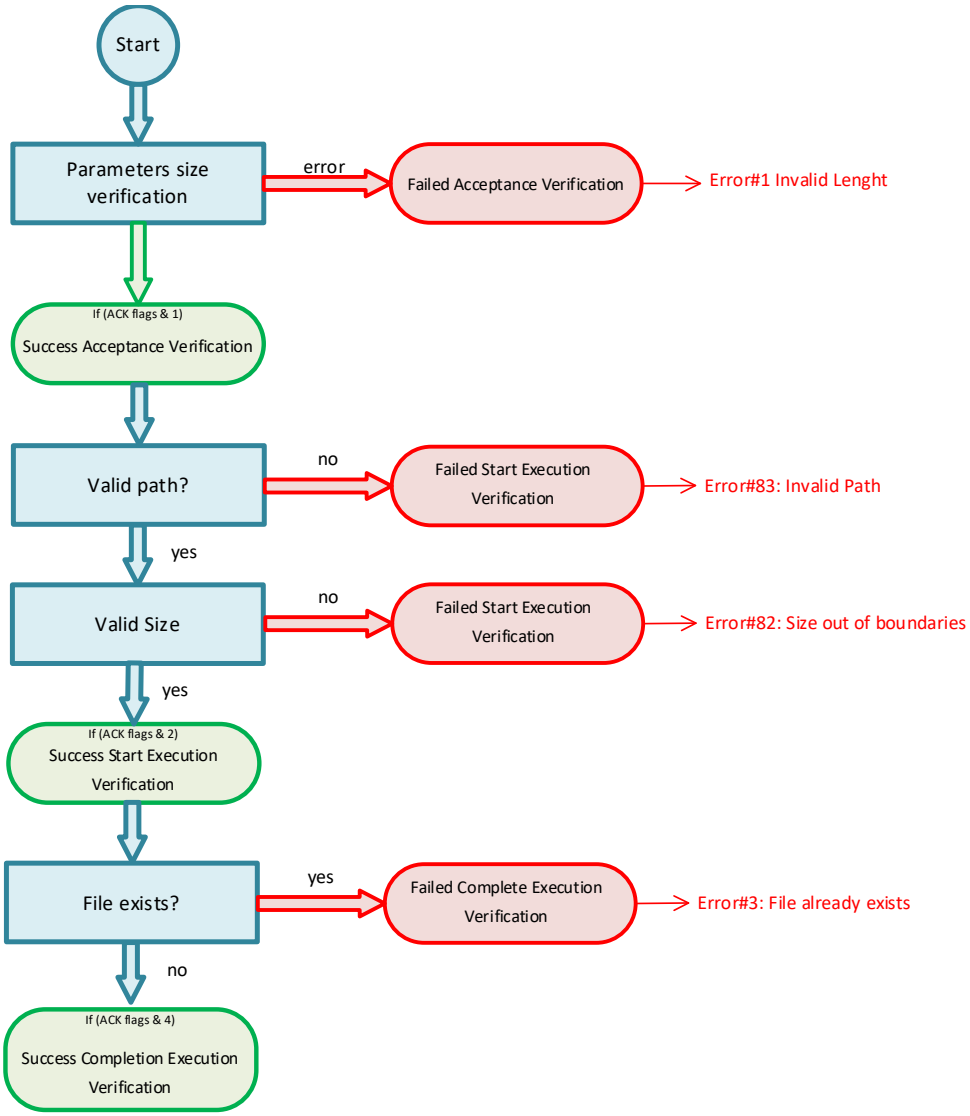
The message report output

Subtypes requests

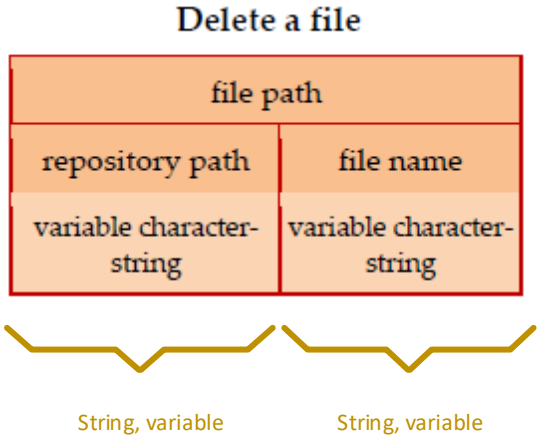
TC[23,1] create a file



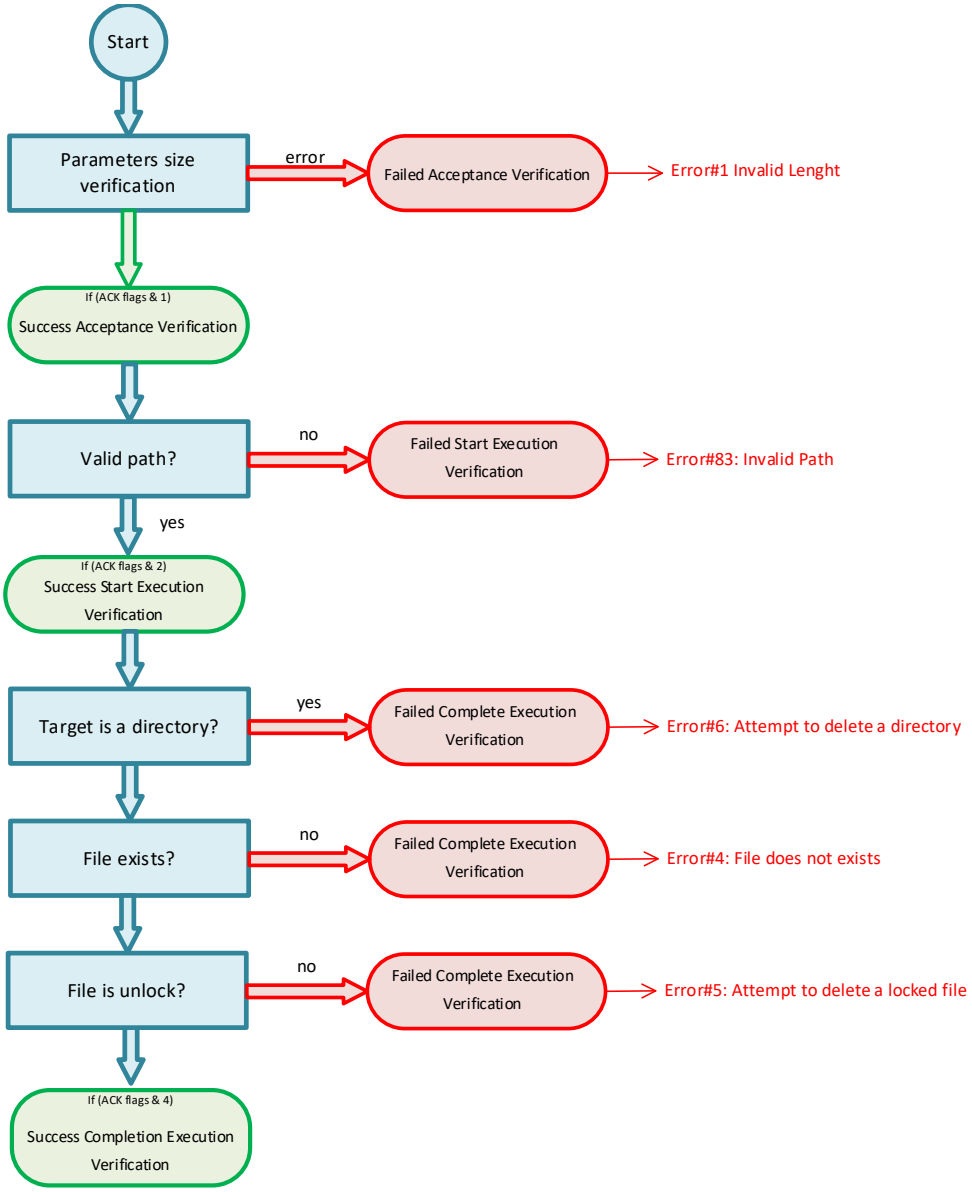
Message request verification flow



TC[23,2] delete a file


Delete a file
file path
repository path
file name
variable character-string
variable character-string
String, variable
String, variable

Message request verification flow



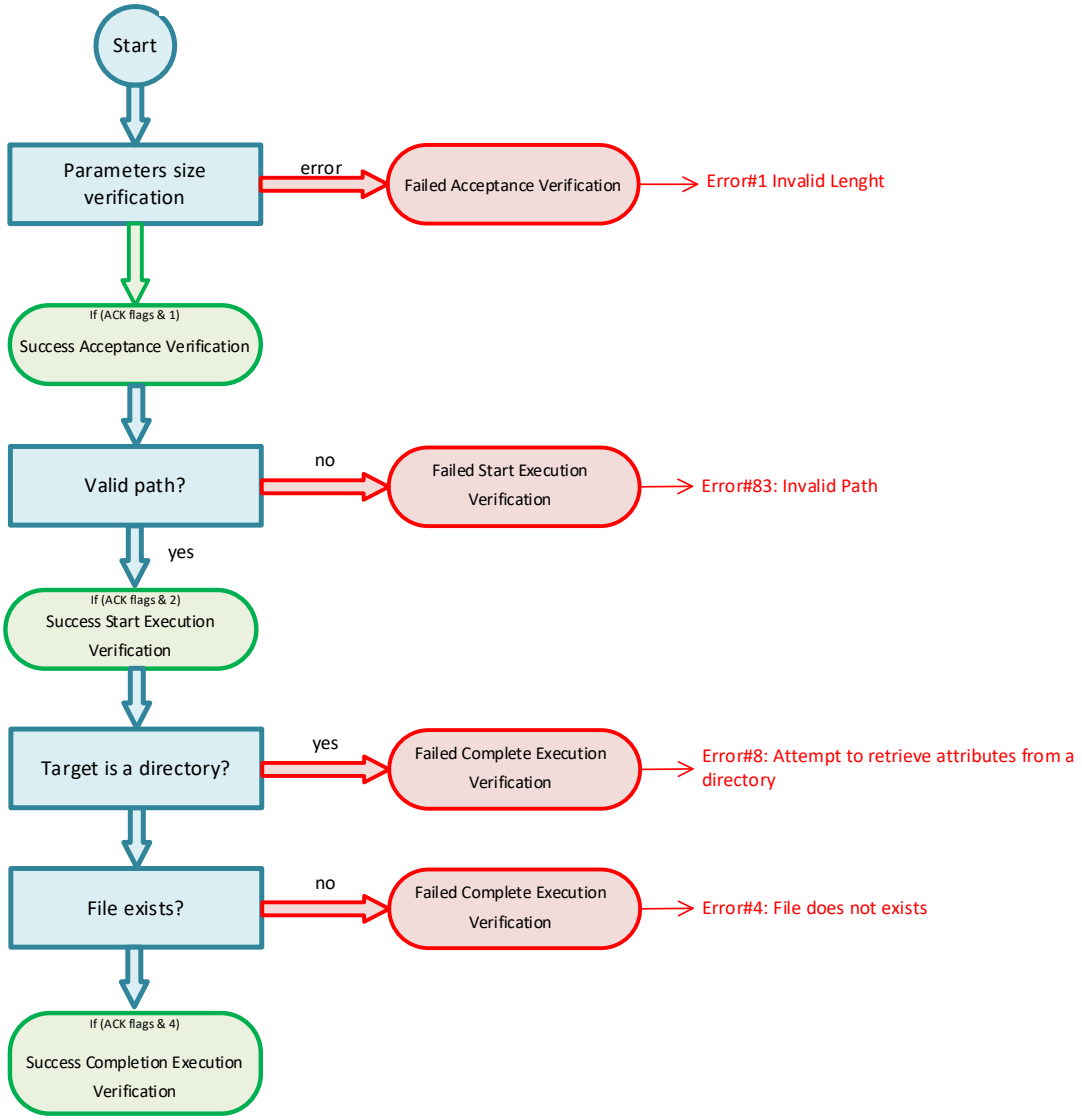
TC[23,3] report the attributes of a file

Report the attributes of a file

file path	
repository path	file name
variable character-string	variable character-string

String, variable String, variable

Message request verification flow



TM[23,4] file attribute report

File attribute report

repeated N times

file path		file size	file locked status	additional file attributes
repository path	file name			
variable character-string	variable character-string	unsigned integer	Boolean	deduced
			optional	*optional*

String, variable String, variable uint32 uint8

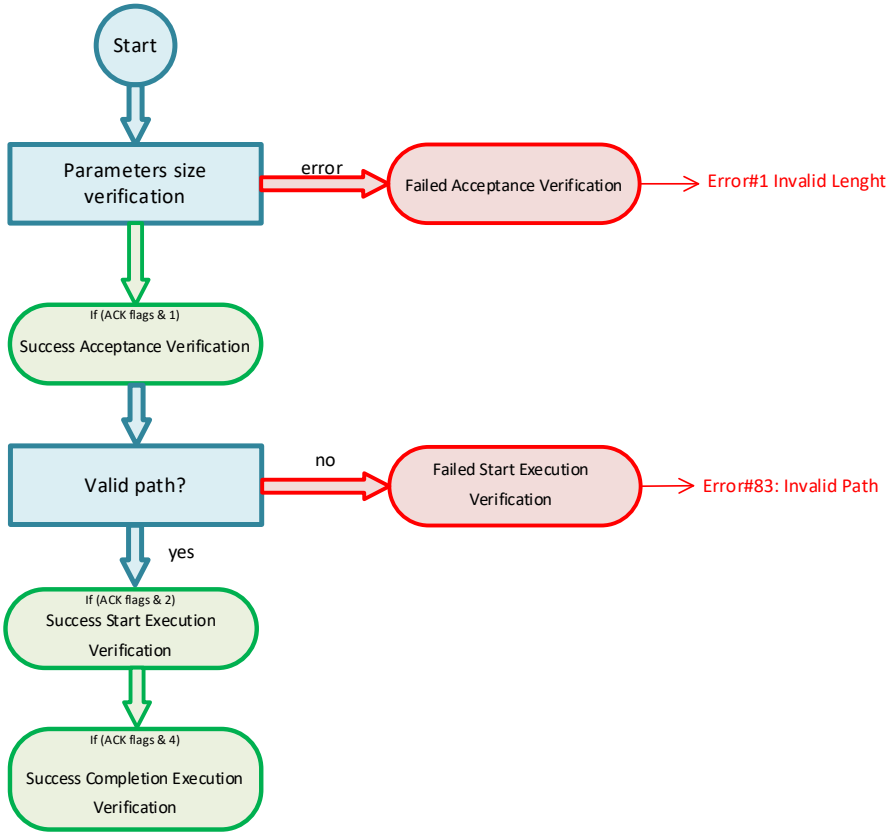
TC[23,5] lock a file

Lock a file

file path	
repository path	file name
variable character-string	variable character-string

String, variable String, variable

Message request verification flow



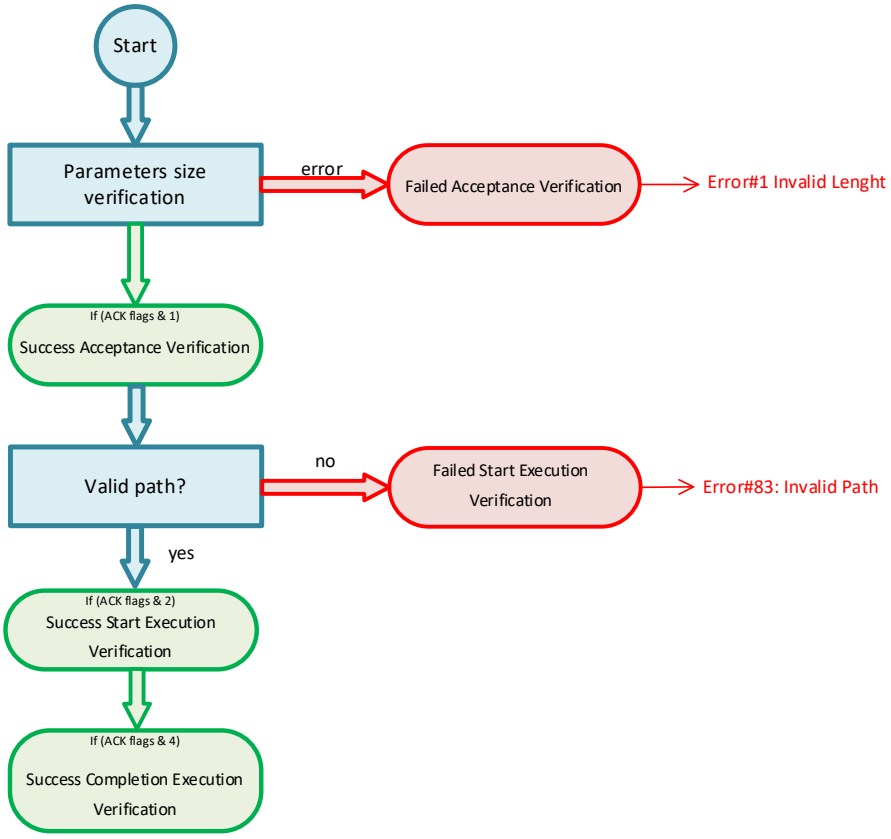
TC[23,6] unlock a file

Unlock a file

file path	
repository path	file name
variable character-string	variable character-string

String, variable String, variable

Message request verification flow



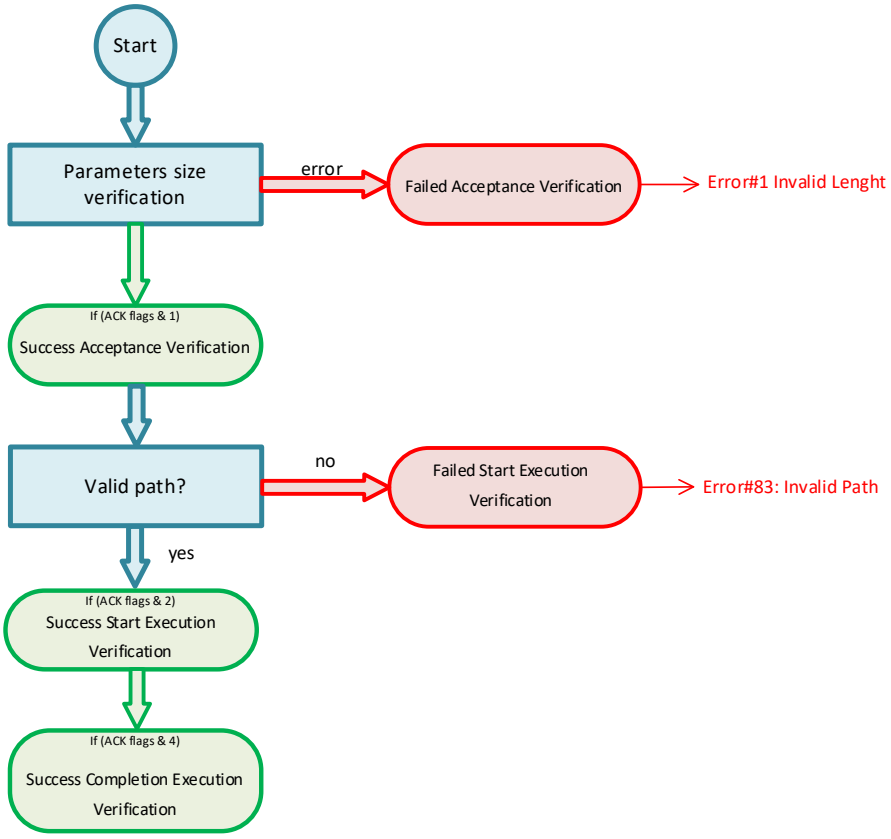
TC[23,7] find files

Find files

repository path	search pattern
variable character-string	variable character-string

String, variable String, variable

Message request verification flow



TM[23,8] found files report

Found files report

repeated N times

repository path	search pattern	N	matching file path
variable character-string	variable character-string	unsigned integer	variable character-string

String, variable | String, variable | uint16 | String, variable

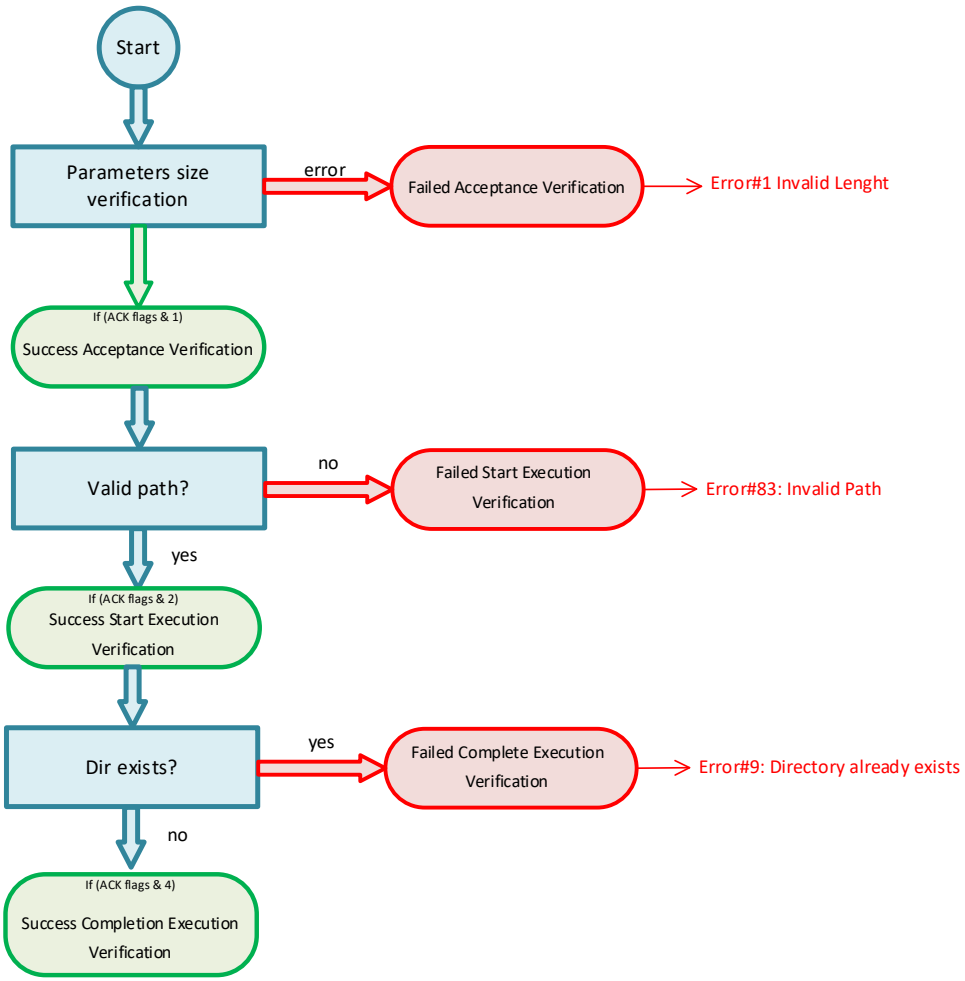
TC[23,9] create a directory

Create a directory

directory path	
repository path	directory name
variable character-string	variable character-string

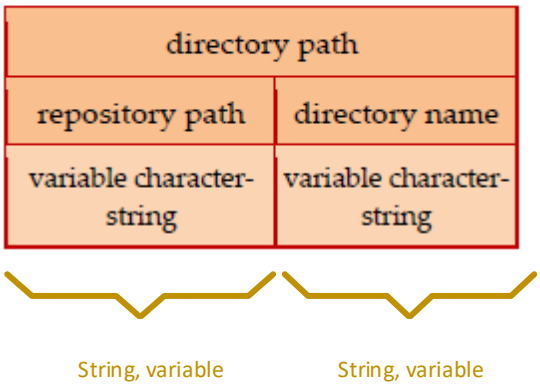
String, variable String, variable

Message request verification flow

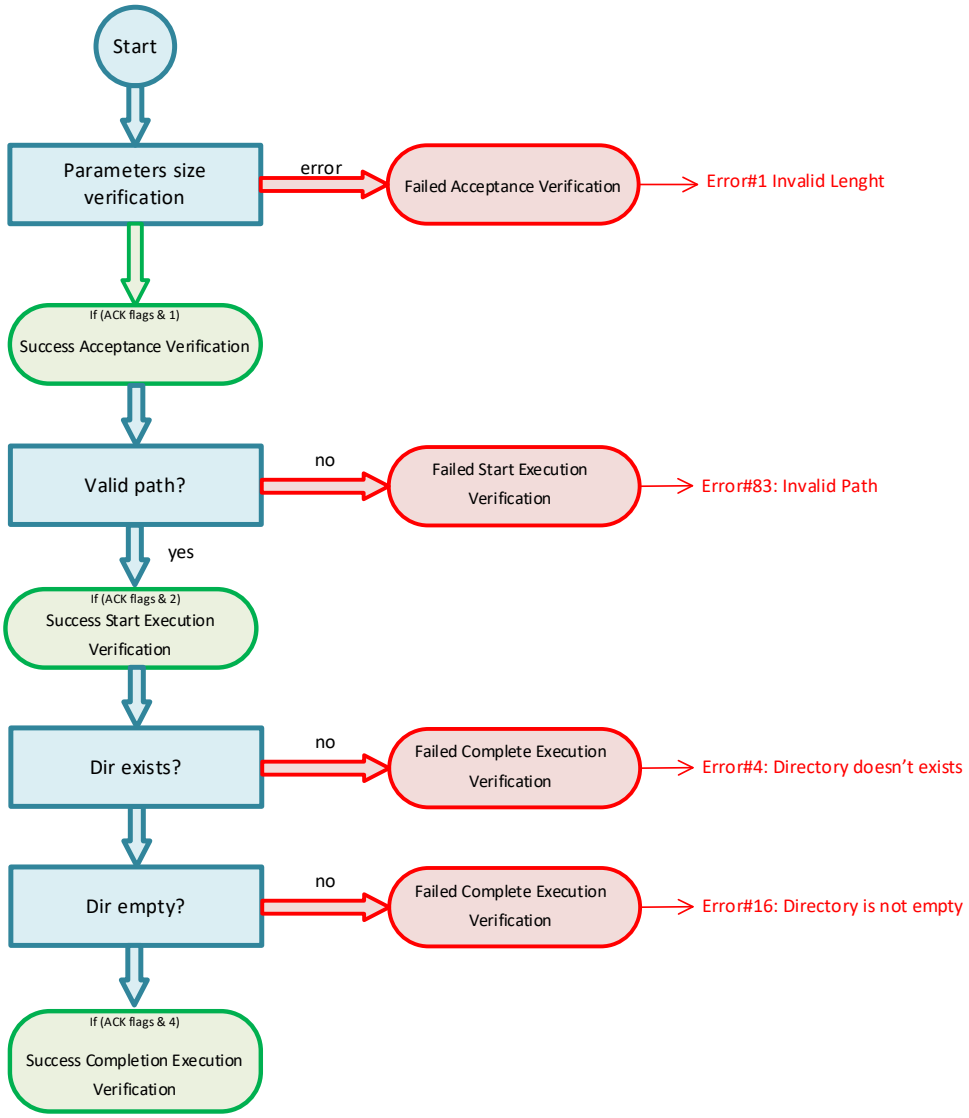


TC[23,10] delete a directory

Delete a directory



Message request verification flow



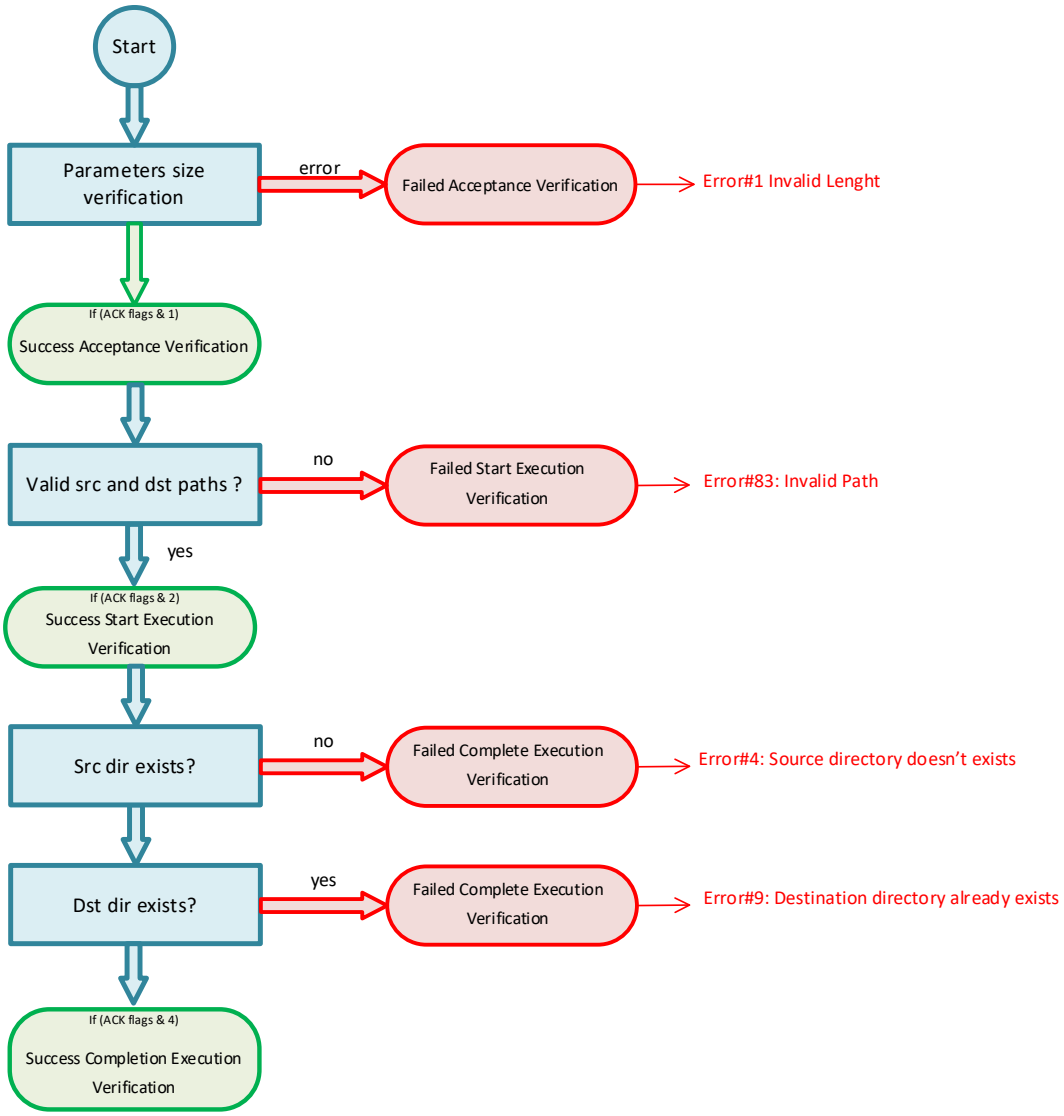
TC[23,11] rename a directory

Rename a directory

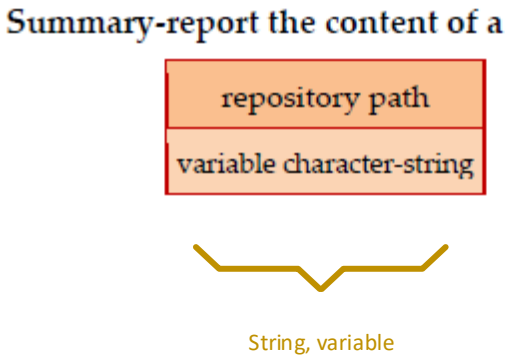
repository path	old directory name	new directory name
variable character-string	variable character-string	variable character-string

String, variable String, variable String, variable

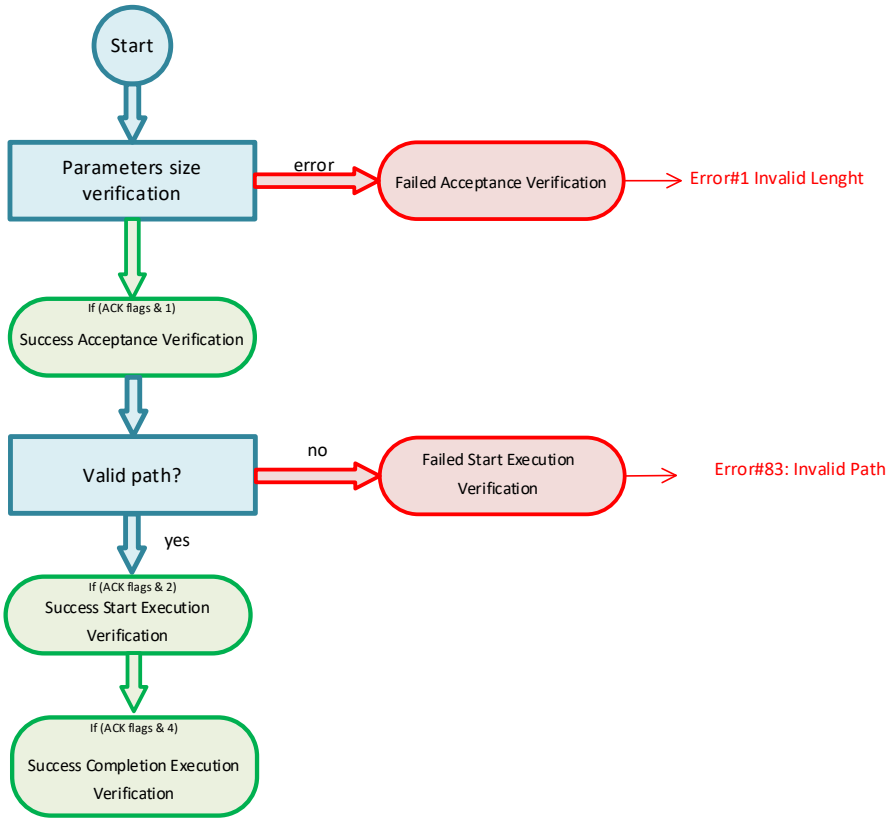
Message request verification flow



TC[23,12] summary-report the content of a repository



Message request verification flow



TM[23,13] repository content summary report

Repository content summary report

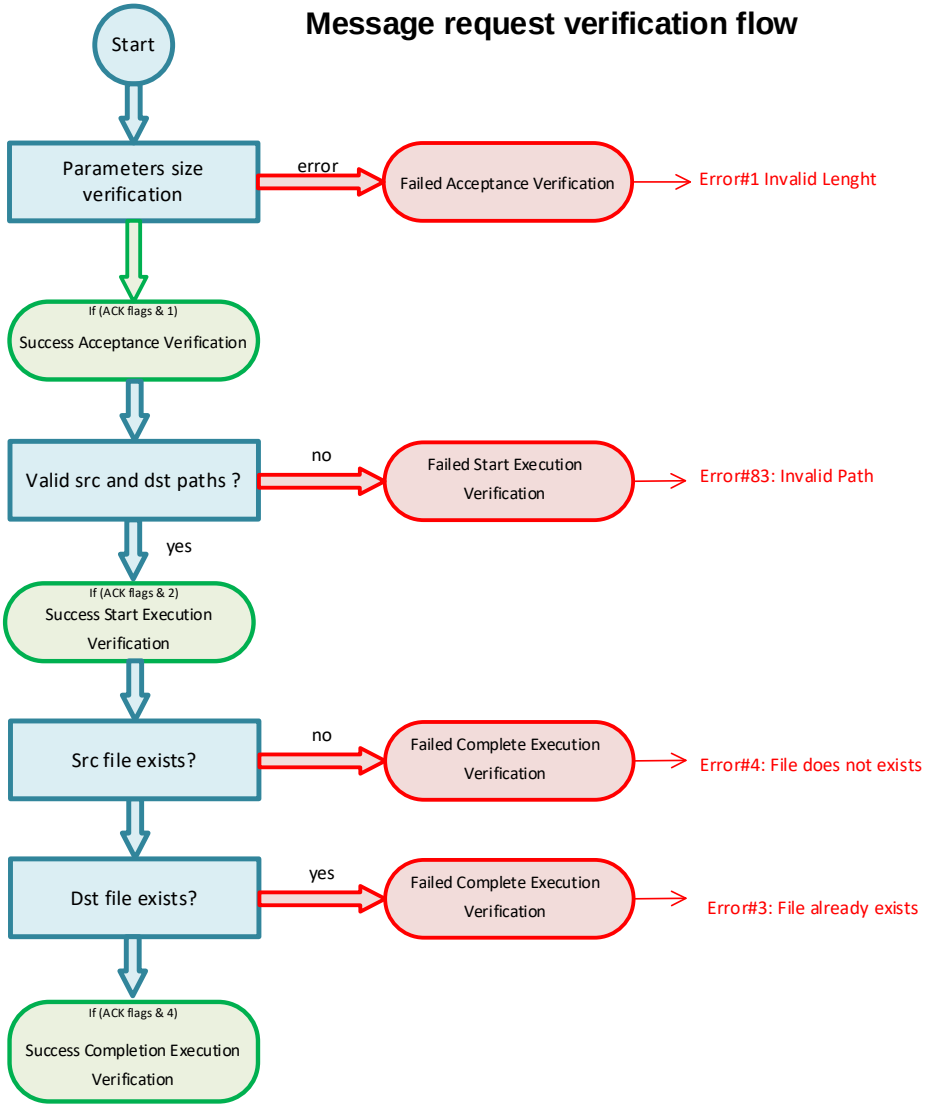
repeated N times

repository path	N	object type	object name
variable character-string	unsigned integer	enumerated	variable character-string
String, variable	uint16	uint8	String, variable

TC[23,14] copy a file

Copy a file

operation ID	source file path		target file path	
	repository path	file name	repository path	file name
unsigned integer	variable character-string	variable character-string	variable character-string	variable character-string
uint16	String, variable	String, variable	String, variable	String, variable

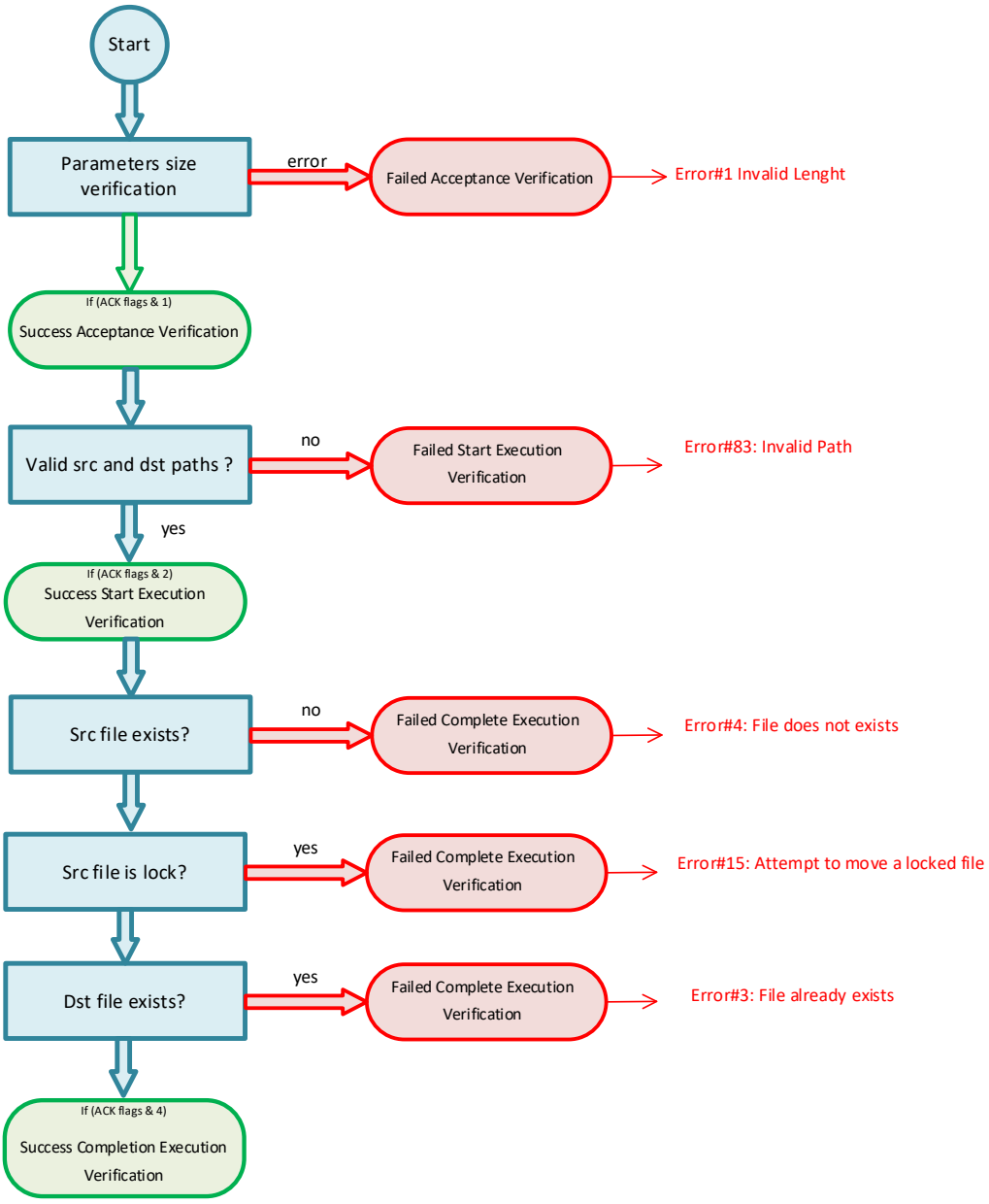


TC[23,15] move a file

Move a file

operation ID	source file path		target file path	
	repository path	file name	repository path	file name
unsigned integer	variable character-string	variable character-string	variable character-string	variable character-string
uint16	String, variable	String, variable	String, variable	String, variable

Message request verification flow



3 INSTALLATION

Install the Embed Template Library:

```
git clone https://github.com/ETLCPP/etl.git
cd etl
git checkout <targetVersion>
cmake -B build .
sudo cmake --install build/
```

Install pySerial for the serial port block helper

```
pip3 install pyserial
```

or

```
sudo apt-get update -y
sudo apt install python3-serial
```

Install nlohmann for json parsing:

```
sudo apt-get update –y
sudo apt-get install -y nlohmann-json-dev
```

Then install gr-pus:

```
git clone https://github.com/gjg/gr-pus.git
cd gr-pus
mkdir build
cd build
cmake ..
make
sudo make install
sudo ldconfig
```

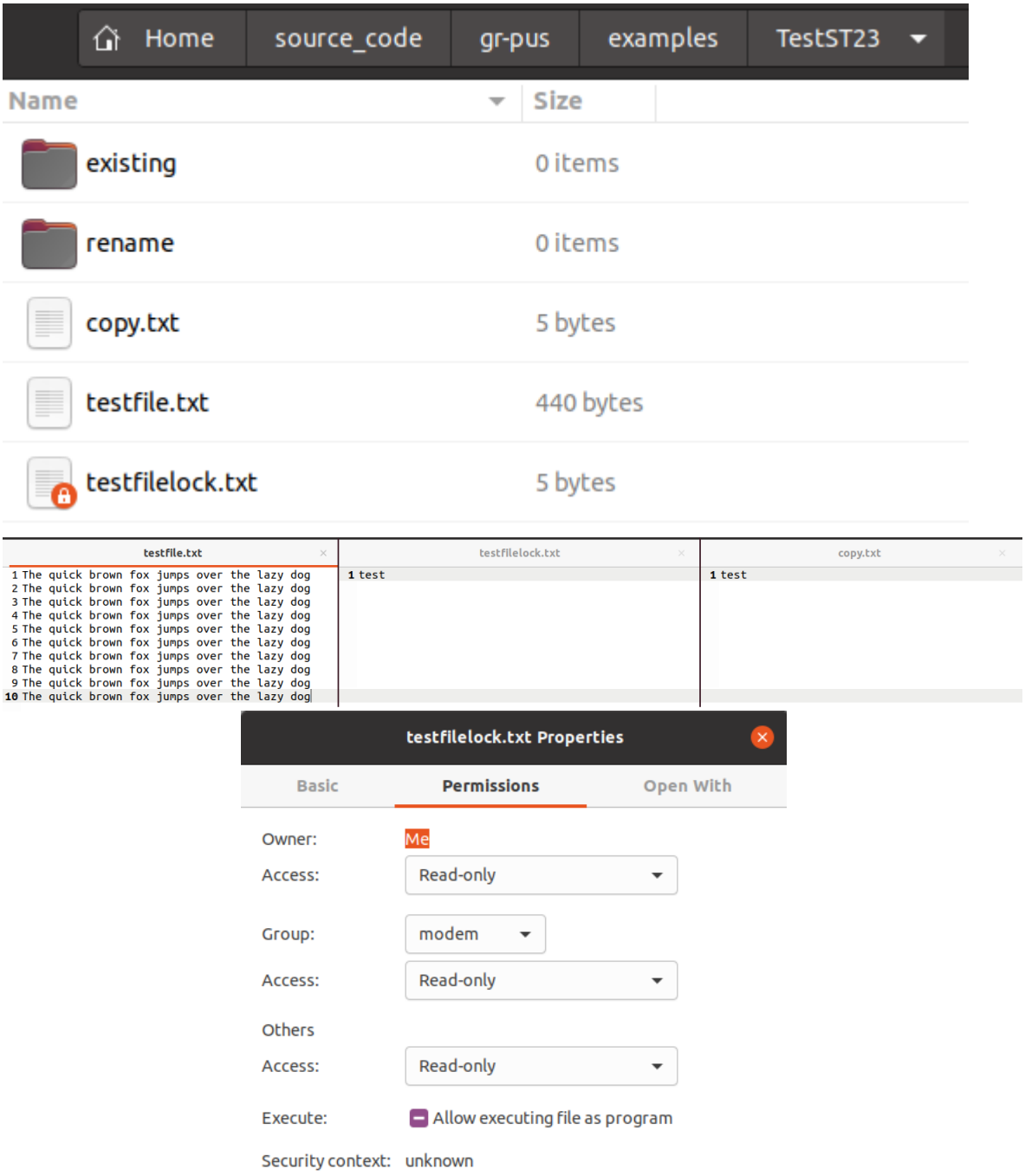
Review the file

gr-pus/include/gnuradio/pus/Definitions/ECSS_Definitions.h

It contains the gr-pus configuration, check if the size constraints will meet your need otherwise change them before compiling (execute make)

The gr-pus/examples folder has an example flowgraph and its json files. These json files, also, and the subfolder called TestST23 are used for testing, please don't change them if you are planning to run the qa tests

Before running the test, ensures the proper TestST23 folder configuration:



4 BINDING SPECIFIC APPLICATION WITH THIS IMPLEMENTATION

4.1 MESSAGE'S INTERFACES

The pus_example.grc in examples directory uses a Serial Transceiver OOT block whitch send/receive data to/from a UART serial port, but any in/out interface could

by used to receive the message requests and to send the message reports, additional interfaces could be required to send the forwarded messages and the stored ones

All messages are byte vectors and no metadata is used/required (pmt::PMT_NIL)

Next code is the basic one to receive byte vector messages:

```
void class_name::handle_msg(pmt::pmt_t pdu)
{
    // make sure PDU data is formed properly
    if (!(pmt::is_pair(pdu))) {
        GR_LOG_NOTICE(d_logger, "received unexpected PMT (non-pair)");
        return;
    }

    pmt::pmt_t meta = pmt::car(pdu);
    pmt::pmt_t v_data = pmt::cdr(pdu);

    // extract data
    if (pmt::is_u8vector(v_data)) {
            std::vector<uint8_t> inData = pmt::u8vector_elements(v_data);

                        // YOUR CODE, the message is in the vector inData

    } else {
            GR_LOG_WARN(d_logger, "Error: the input data is not a u8vector");
    }
 }
```

4.2 PARAMETERS

The Parameters shall be binded againts the real variables, see the SetParameter OOT block code as example in how to do that. Any variable could be binded with a parameter, either a physical variables (ie: temperature) or a soft variable (ie: a GNUradio setting), but the definition of all those variables shall be included into the json file to be load at start up

4.3 MEMORY

This is required if the service ST[06] Memory Managment is used, see the MemoryManager.h/.c code as example in how to do that. You should modify these code to suit your needs

4.4 HARDWARE

The Device Access Service ST[02] has been not implemented because is application dependand (even more than memory management service), then if you need to handle any hardware, you will need to implement your own ST[02] service, use the other services code as reference for this implementation.
Remember for each service at start up shall:

- Get the MessageParser and ErrorHandler singletons
- Get its service type number (for Device Access service type = 2)
- Init the messages counters for each message type to 0
- Register the in/out ports
- Register itself into the messages types list

The code does this is next one:

```
{
    d_message_parser = MessageParser::getInstance();
    d_error_handler = ErrorHandler::getInstance();
    serviceType = ServiceType;

    for(size_t i = 0; i < YOURSERVICE_impl::MessageType::end; i++)
          counters[i] = 0;

    this->message_port_register_in(PMT_IN);
    this->set_msg_handler(PMT_IN,
               [this](pmt::pmt_t msg) { this->handle_msg(msg); });
    this->message_port_register_out(PMT_OUT);
    this->message_port_register_out(PMT_VER);

    std::vector<uint8_t> STMessages;

    for(const auto e : All)
          STMessages.push_back(e);

    AllMessageTypes::MessagesOfService[ServiceType] = STMessages;
}
```

5 KNOWN ISSUES

Issues

- In the example flowgraph, most of the message request vector associated with GUI buttons haven't the proper configuration (size, CRC, etc)
- For periodic events testing, as housekeeping report generation, request sequencing, etc, the test code wait a few seconds for sync events upon test needs, but because the timer tick start when a second start, then, one or

two of the test steps could fail, the revision of those specific cases is pending in this release

- QA Test files shared no code, each one is self cointained, extract and unify common code is pending in this release
- Code's warnings cleaning is pending for this release
- The "gr-pus description and user manual" (this document) has basic information, but is incomplete in this release