PILOTS for robotic Inspection and maintenance Grounded on advanced intelligent platforms and prototype applications

Deliverable DX.X

gRCS – Interface with the Robotic Vehicle

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 871542

Project Information

Grant Agreement Number	ICT-871542
Starting Date	1st January, 2020
Duration	48 months
Call Identifier	H2020-ICT-2018-2020
Topic	ICT-09-2019-2020 (B) (IA)
Project Website	www.piloting-project.eu
Project Manager	Antidio Viguria
Organisation	Fundación Andaluza para el Desarrollo Aeroespacial (CATEC)
E-mail	aviguria@catec.aero

Document Information

Due date:	**Submission date:**
Lead beneficiary:	**Version:**
Work Package	
Deliverable Title	
Main Editor(s)	
Contributor(s)	
Reviewer (s)	

Document Clasification					
Status	D	**Nature**	R	**Dissemination level**	CO
Draft: D Final: F		R = Report P = Prototype D = Demonstrator O = Other		PU = Public PP = Restricted to other programme participants* RE = Restricted to a group specified by the consortium CO = Confidential, only for members of the consortium * *including the Commission Services	

Revision history			
Version	**Issue Date**	**Status**	**Autor(s)**
0.1	18/12/2020	First Draft	CATEC
1.0	18/01/2021	First Version	CATEC

2.0	26/03/2021	Second Version	CATEC

Contents

List of Figures

List of Tables

Acronyms

Abbreviation	Explanation

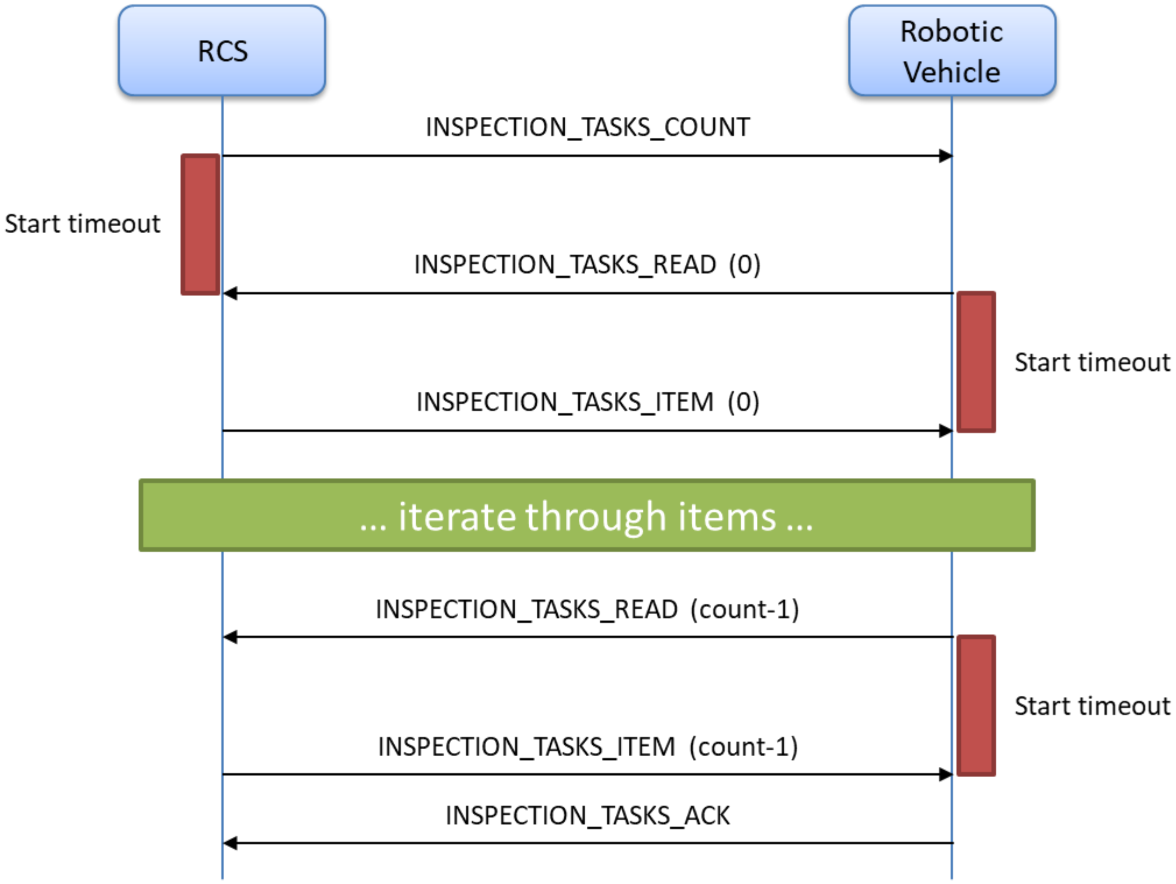
Table 1. Acronyms list

1. Communication sequences

1.1 Upload the inspection tasks to the Robotic Vehicle

Before starting a mission, the gRCS has to send the inspection task list to the robotic vehicle.

The diagram below shows the communication sequence to upload a new inspection tasks (waypoint list) to the robotic vehicle. This diagram assumes that all operations are performed successfully.


RCS
Robotic Vehicle
INSPECTION_TASKS_COUNT
Start timeout
INSPECTION_TASKS_READ (0)
Start timeout
INSPECTION_TASKS_ITEM (0)
... iterate through items ...
INSPECTION_TASKS_READ (count-1)
Start timeout
INSPECTION_TASKS_ITEM (count-1)
INSPECTION_TASKS_ACK

In more detail, the sequence of operations is:

1. The gRCS sends INSPECTION_TASKS_COUNT message (see section 2.1.1) including the number of inspection tasks items to be uploaded (count).
    a. A timeout must be started for the gRCS to wait on the INSPECTION_TASKS_READ (see section 2.1.2) response from robotic vehicle.
    b. If the timeout expires without a response being received, then the INSPECTION_TASKS_COUNT message must be resent.

2. The robotic vehicle receives message and responds with INSPECTION_TASKS_READ message requesting the first inspection tasks item (seq==0).
    a. A timeout must be started for the robotic vehicle to wait on the INSPECTION_TASKS_ITEM (see section 2.1.3) response from the gRCS.
    b. If the timeout expires without a response being received, then the INSPECTION_TASKS_READ message must be resent.
3. The gRCS receives INSPECTION_TASKS_READ message and responds with the requested inspection tasks item in an INSPECTION_TASKS_ITEM message.
4. The robotic vehicle and the gRCS repeat this INSPECTION_TASKS _READ / INSPECTION_TASKS_ITEM cycle, iterating seq until all items are uploaded (seq==count-1).
5. After receiving the last inspection tasks item, the robotic vehicle responds with INSPECTION_TASKS_ACK message (see section 2.1.4) with the type of INSPECTION_TASKS_ACCEPTED (see section 2.2.3) indicating that the inspection tasks upload was completed successfully.
    a. The robotic vehicle should set the new data received to be the current inspection tasks, discarding the original data.
    b. The robotic vehicle considers the upload complete.
6. The gRCS receives INSPECTION_TASKS_ACK message containing INSPECTION_TASKS_ACCEPTED to indicate the operation is complete.

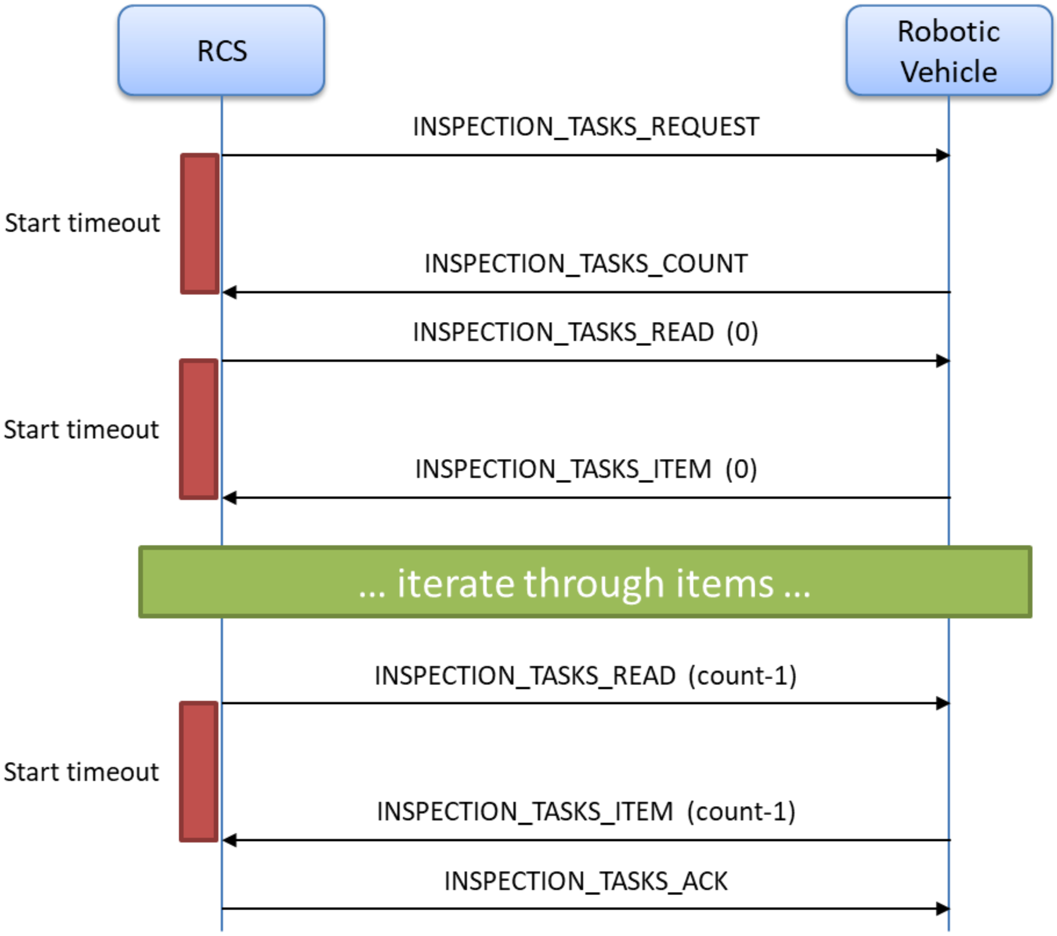
Note:

- A timeout is set for every message that requires a response. If the timeout expires without a response being received, then the request must be resent.
- Inspection tasks items must be received in order. If an item is received out-of-sequence the expected item should be re-requested by the vehicle (the out-of-sequence item is dropped).
- An error can be signaled in response to any request using an INSPECTION_TASKS_ACK message containing an error code. This must cancel the operation and restore the inspection tasks to its previous state. For example, the robotic vehicle might respond to the INSPECTION_TASKS_COUNT request with an INSPECTION_TASKS_NO_SPACE if there is not enough space to upload the inspection tasks.
- Uploading an empty inspection tasks (INSPECTION_TASKS_COUNT is 0) has the effect of clearing the current inspection tasks.

1.2 Download the inspection tasks from the Robotic Vehicle

During the execution of the mission, it may be the case that the robotic vehicle does a re-planning of the inspection task list (for example, when an obstacle is encountered along the way). If this occurs, the gRCS has to request the modified inspection task list from the robotic vehicle.

The diagram below shows the communication sequence to download the current inspection tasks from the robotic vehicle (assuming all operations succeed).


RCS
Robotic Vehicle
INSPECTION_TASKS_REQUEST
Start timeout
INSPECTION_TASKS_COUNT
INSPECTION_TASKS_READ (0)
Start timeout
INSPECTION_TASKS_ITEM (0)
... iterate through items ...
INSPECTION_TASKS_READ (count-1)
Start timeout
INSPECTION_TASKS_ITEM (count-1)
INSPECTION_TASKS_ACK

The sequence is like that for uploading the inspection tasks. The main difference is that the gRCS sends INSPECTION_TASKS_REQUEST message (see section 2.1.5), which triggers the robotic vehicle to respond with the current count of items. This starts a cycle where the gRCS requests inspection tasks items, and the robotic vehicle supplies them.

Note:

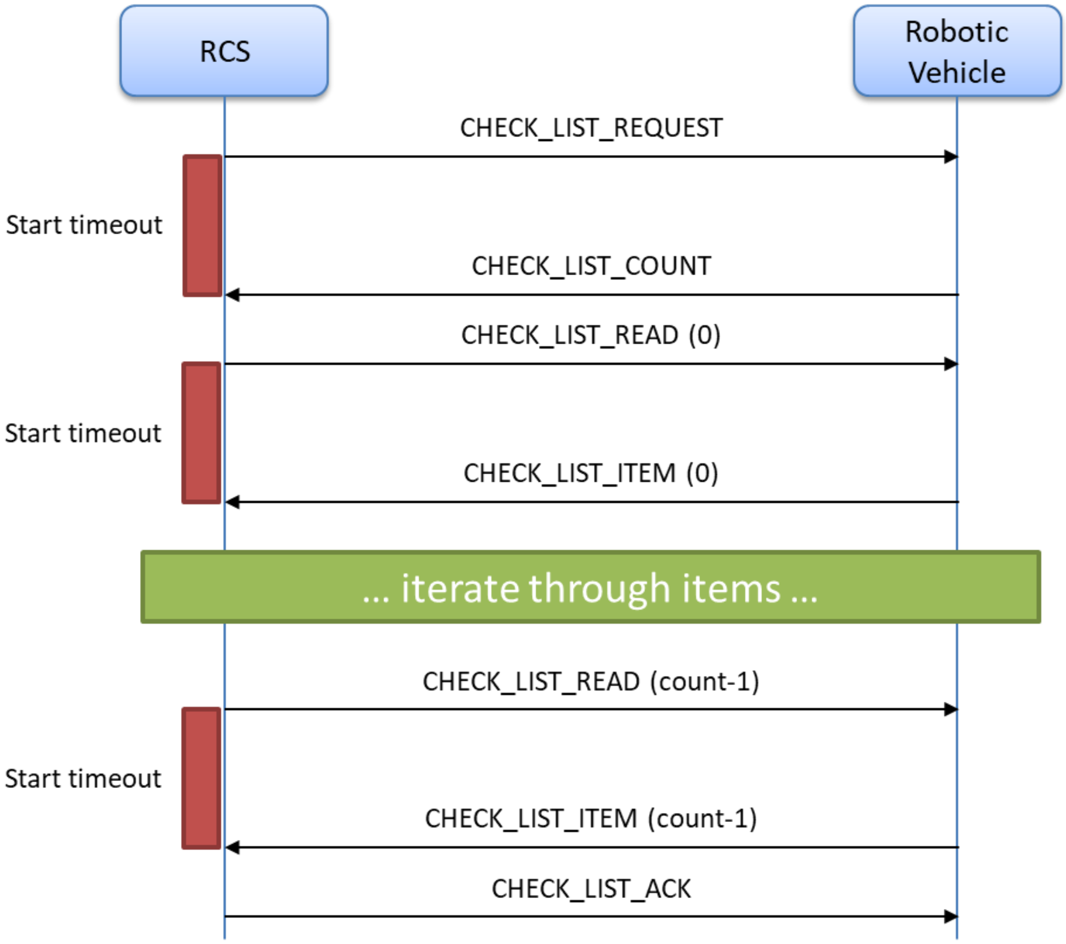
- A timeout is set for every message that requires a response. If the timeout expires without a response being received, then the request must be resent.
- Inspection tasks items must be received in order. If an item is received out-of-sequence the expected item should be re-requested by the gRCS (the out-of-sequence item is dropped).

- An error can be signaled in response to any request using an INSPECTION_TASKS_ACK message containing an error code. This must cancel the operation.

1.3 Download the checklist from the Robotic Vehicle

Before starting the mission, the gRCS operator has to perform a set of checks associated with the robotic vehicle. To do this, the gRCS must have previously asked the robotic vehicle for its checklist.

The diagram below shows the communication sequence to download the checklist from the robotic vehicle (assuming all operations succeed).



In more detail, the sequence of operations is:

1. The gRCS sends CHECK_LIST_REQUEST message (see section 2.1.6) requesting the robotic vehicle checklist.
    a. A timeout must be started for the gRCS to wait on the CHECK_LIST_COUNT (see section 2.1.7) response from robotic vehicle.

   b. If the timeout expires without a response being received, then the CHECK_LIST_REQUEST message must be resent.

2. The robotic vehicle receives message and responds with CHECK_LIST_COUNT message including the number of check items to be downloaded (count).

3. The gRCS receives message and responds with CHECK_LIST_READ message (see section 2.1.8) requesting the first check item (seq==0).

   a. A timeout must be started for the gRCS to wait on the CHECK_LIST_ITEM (see section 2.1.9) response from the robotic vehicle.

   b. If the timeout expires without a response being received, then the CHECK_LIST_READ message must be resent.

4. The robotic vehicle receives CHECK_LIST_READ message and responds with the requested check item in a CHECK_LIST_ITEM message.

5. The gRCS and the robotic vehicle repeat the CHECK_LIST_READ / CHECK_LIST_ITEM cycle, iterating seq until all items are downloaded (seq==count-1).

6. After receiving the last check item, the gRCS responds with a CHECK_LIST_ACK message (see section 2.1.10) with the type of CHECK_LIST_ACCEPTED (see section 2.2.4) indicating that the checklist download was completed successfully.

   a. The gRCS should save the checklist received, discarding any lists it had previously.

   b. The gRCS considers the download complete.

7. The robotic vehicle receives CHECK_LIST_ACK message containing CHECK_LIST_ACCEPTED to indicate the operation is complete.

Note:

- A timeout is set for every message that requires a response. If the timeout expires without a response being received, then the request must be resent.

- Check items must be received in order. If an item is received out-of-sequence the expected item should be re-requested by the gRCS (the out-of-sequence item is dropped).

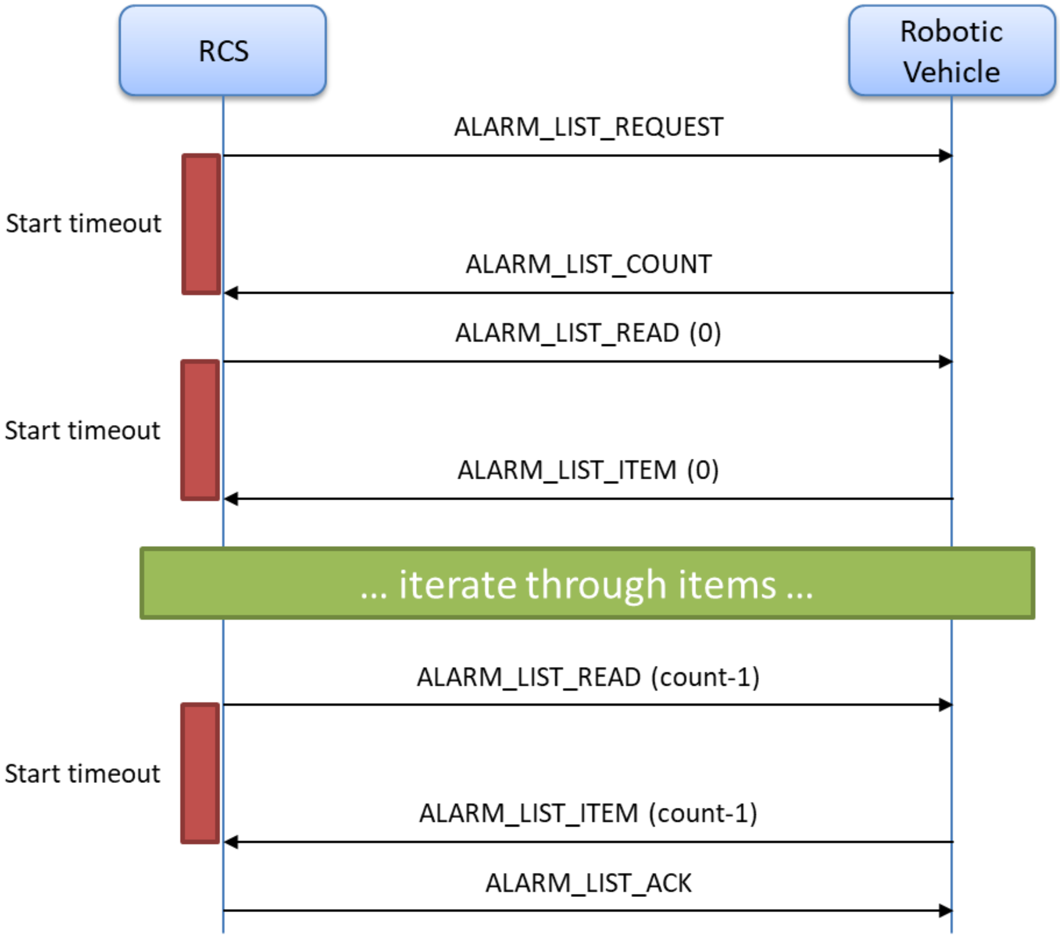
- An error can be signaled in response to any request using a CHECK_LIST_ACK message containing an error code. This must cancel the operation and restore the checklist inside of the gRCS to its previous state.

- Downloading an empty checklist (CHECK_LIST_COUNT is 0) has the effect of clearing the current checklist saved inside of the gRCS.

1.4 Download the list of alarms from the Robotic Vehicle

During the execution of the mission, the gRCS can show the operator the status of a set of alarms predefined by the robotic vehicle (for example, the status of its batteries). To do this, the gRCS must have previously requested the robotic vehicle for its alarm list.

The diagram below shows the communication sequence to download the list of alarms from the robotic vehicle (assuming all operations succeed).



In more detail, the sequence of operations is:

1. The gRCS sends ALARM_LIST_REQUEST message (see section 2.1.11) requesting the robotic vehicle alarm list.

    a. A timeout must be started for the gRCS to wait on the ALARM_LIST_COUNT (see section 2.1.12) response from robotic vehicle.

   b. If the timeout expires without a response being received, then the ALARM_LIST_REQUEST message must be resent.

2. The robotic vehicle receives message and responds with ALARM_LIST_COUNT message including the number of alarm items to be downloaded (count).

3. The gRCS receives message and responds with ALARM_LIST_READ message (see section 2.1.13) requesting the first alarm item (seq==0).

   a. A timeout must be started for the gRCS to wait on the ALARM_LIST_ITEM (see section 2.1.14) response from the robotic vehicle.

   b. If the timeout expires without a response being received, then the ALARM_LIST_READ message must be resent.

4. The robotic vehicle receives ALARM_LIST_READ message and responds with the requested alarm item in an ALARM_LIST_ITEM message.

5. The gRCS and robotic vehicle repeat the ALARM_LIST_READ / ALARM_LIST_ITEM cycle, iterating seq until all items are downloaded (seq==count-1).

6. After receiving the last alarm item, the gRCS responds with an ALARM_LIST_ACK message (see section 2.1.15) with the type ALARM_LIST_ACCEPTED (see section 2.2.5) indicating that the download of the alarm list was completed successfully.

   a. The gRCS should save the alarm list received, discarding any lists it had previously.

   b. The gRCS considers the download complete.

7. The robotic vehicle receives ALARM_LIST_ACK message containing ALARM_LIST_ACCEPTED to indicate the operation is complete.

Note:

- A timeout is set for every message that requires a response. If the timeout expires without a response being received, then the request must be resent.

- Alarm items must be received in order. If an item is received out-of-sequence the expected item should be re-requested by the gRCS (the out-of-sequence item is dropped).

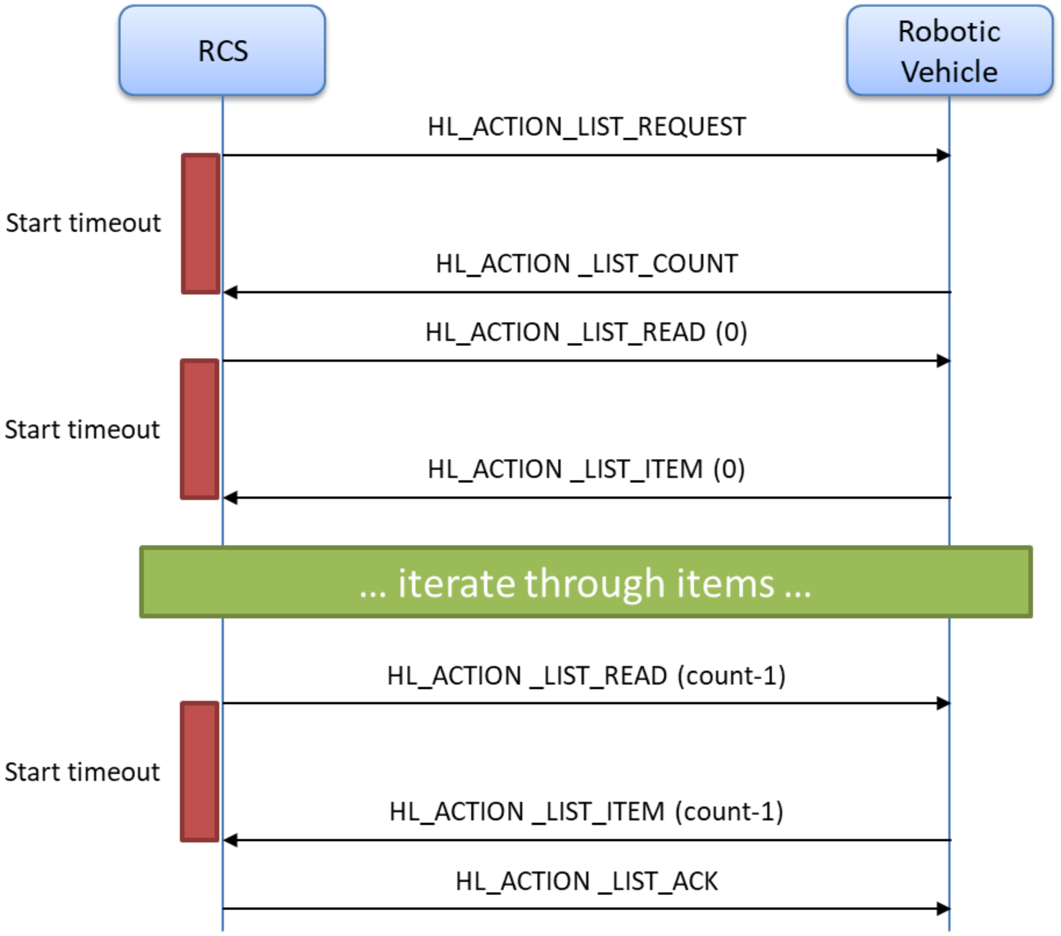
- An error can be signaled in response to any request using an ALARM_LIST_ACK message containing an error code. This must cancel the operation and restore the alarm list inside of the gRCS to its previous state.

- Downloading an empty alarm list (ALARM_LIST_COUNT is 0) has the effect of clearing the current alarm list saved inside of the gRCS.

1.5 Download the list of high-level actions from the Robotic Vehicle

During the execution of the mission, the gRCS operator can command the robotic vehicle a set of high-level actions. To do this, the gRCS must have previously asked the robotic vehicle for its high-level action list.

The diagram below shows the communication sequence to download the list of high-level actions from the robotic vehicle (assuming all operations succeed).



In more detail, the sequence of operations is:

1. The gRCS sends HL_ACTION_LIST_REQUEST message (see section 2.1.16) requesting the robotic vehicle high-level action list.
    a. A timeout must be started for the gRCS to wait on the HL_ACTION_LIST_COUNT (see section 2.1.17) response from robotic vehicle.

   b. If the timeout expires without a response being received, then the HL_ACTION_LIST_REQUEST message must be resent.

2. The robotic vehicle receives message and responds with HL_ACTION_LIST_COUNT message including the number of high-level action items to be downloaded (count).

3. The gRCS receives message and responds with HL_ACTION_LIST_READ message (see section 2.1.18) requesting the first high-level action item (seq==0).

   a. A timeout must be started for the gRCS to wait on the HL_ACTION_LIST_ITEM (see section 2.1.19) response from the robotic vehicle.

   b. If the timeout expires without a response being received, then the HL_ACTION_LIST_READ message must be resent.

4. The robotic vehicle receives HL_ACTION_LIST_READ message and responds with the requested high-level action item in a HL_ACTION_LIST_ITEM message.

5. The gRCS and robotic vehicle repeat the HL_ACTION_LIST_READ / HL_ACTION_LIST_ITEM cycle, iterating seq until all items are downloaded (seq==count-1).

6. After receiving the last high-level action item, the gRCS responds with an HL_ACTION_LIST_ACK message (see section 2.1.20) with the type HL_ACTION_LIST_ACCEPTED (see section 2.2.6) indicating that the download of the high-level action list was completed successfully.

   a. The gRCS should save the high-level action list received, discarding any lists it had previously.

   b. The gRCS considers the download complete.

7. The robotic vehicle receives HL_ACTION_LIST_ACK message containing HL_ACTION_LIST_ACCEPTED to indicate the operation is complete.

Note:

- A timeout is set for every message that requires a response. If the timeout expires without a response being received, then the request must be resent.

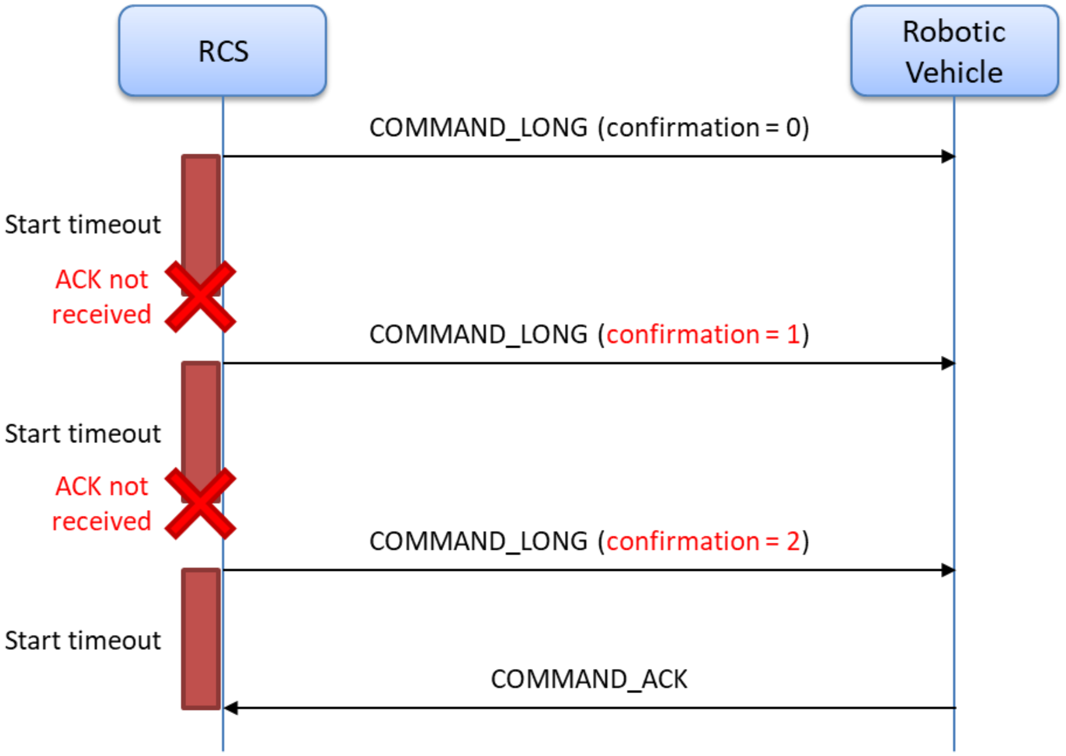
- High-level action items must be received in order. If an item is received out-of-sequence the expected item should be re-requested by the gRCS (the out-of-sequence item is dropped).

- An error can be signaled in response to any request using an HL_ACTION_LIST_ACK message containing an error code. This must cancel the operation and restore the high-level action list inside of the gRCS to its previous state.
- Downloading an empty high-level action list (HL_ACTION_LIST_COUNT is 0) has the effect of clearing the current high-level action list saved inside of the gRCS.

1.6 Send commands to the Robotic Vehicle

During the execution of the mission, the gRCS operator can send commands to the robotic vehicle (for example, the commands associated with high-level actions).

The diagram below shows the communication sequence to send a command to the robotic vehicle.



In more detail, the sequence of operations is:

1. The gRCS sends COMMAND_LONG (see section 2.1.21) that includes the identifier of the command to be executed by the robotic vehicle.
   a. A timeout must be started for the gRCS to wait on the COMMAND_ACK (see section 2.1.22) response from robotic vehicle.
   b. If the timeout expires without a response being received, then the COMMAND_LONG message must be resent, increasing the value of the confirmation field.

2. The robotic vehicle receives message and responds with COMMAND_ACK with the type of MAV_RESULT_ACCEPTED (see section 2.2.2) indicating the command is valid.
    a. The robotic vehicle should execute the command.
3. The gRCS receives COMMAND_ACK containing MAV_RESULT_ACCEPTED to indicate the operation is complete.

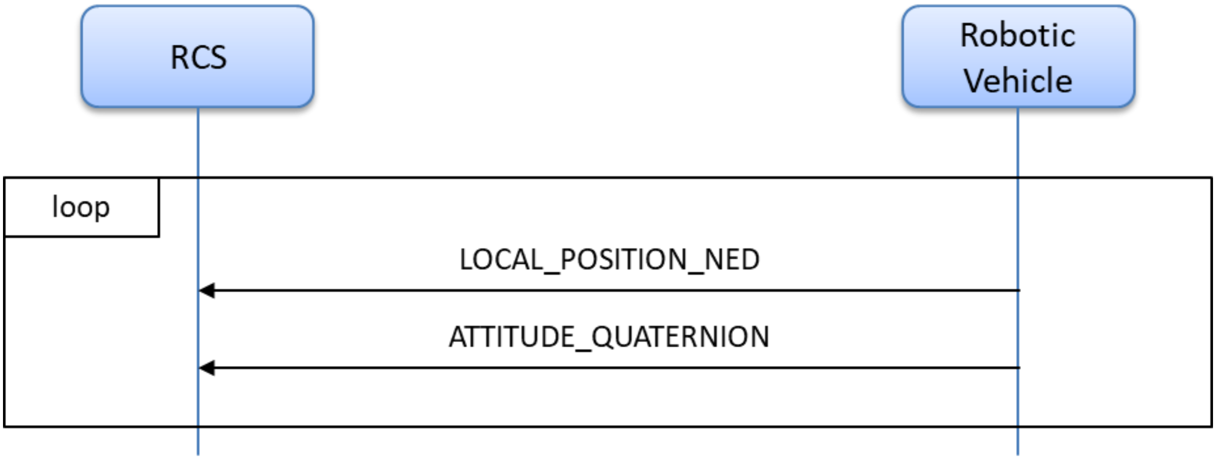
Note:

- A timeout is set for every message that requires a response. If the timeout expires without a response being received, then the request must be resent.
- An error can be signaled in response to any request using a COMMAND_ACK message containing an error code. This must cancel the operation. For example, the robotic vehicle might respond to the COMMAND_LONG request with a MAV_RESULT_FAILED if the command is valid, but the execution has failed.

1.7 Receive pose from the Robotic Vehicle

During the execution of the mission, the robotic vehicle periodically sends its position and orientation to the gRCS. As well as the position and orientation of its sensors.

The diagram below shows the communication sequence to receive the position and the orientation from the robotic vehicle.

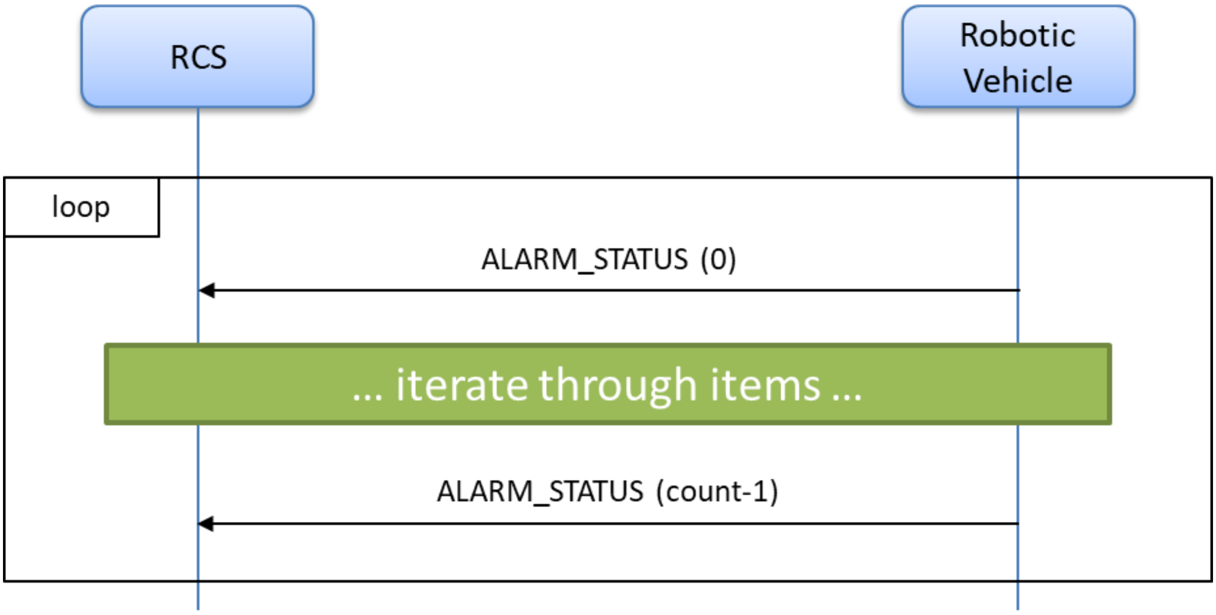


1. The robotic vehicle sends the LOCAL_POSITION_NED (see section 2.1.23) and ATTITUDE_QUATERNION (see section 2.1.24) messages that will contain the position and orientation of the robotic vehicle at each instant of time.
2. The gRCS receives these messages and processes them, without the need to send a confirmation of their receipt.
3. This cycle is repeated periodically until the connection between the robotic vehicle and the gRCS is closed.

1.8 Receive alarm status from the Robotic Vehicle

During the execution of the mission, the robotic vehicle periodically sends to the gRCS the status of the alarms it has defined.

The diagram below shows the communication sequence to receive the alarm status from the robotic vehicle.



1. The robotic vehicle sends an ALARM_STATUS message (see section 2.1.25) for each alarm that it has defined, differentiating them through their identifier.
    a. This message will contain the state of the alarm specified at each instant of time (see section 2.2.7).
2. The gRCS receives these messages and processes them, without the need to send a confirmation of their receipt.
3. This cycle is repeated periodically until the connection between the robotic vehicle and the gRCS is closed.

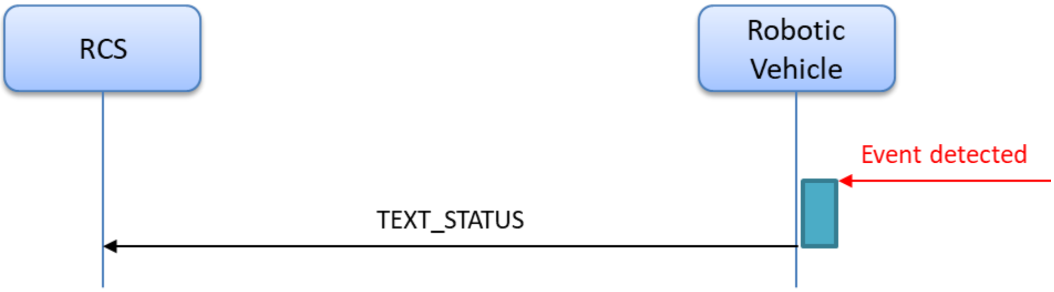
Note:

- The robotic vehicle can send an ALARM_STATUS message at any time without having to wait for the next send cycle to start. So, if an error appears in any of the defined alarms, it can be reported to the gRCS immediately.

1.9 Receive message status from the Robotic Vehicle

During the execution of the mission, the robotic vehicle can send descriptive messages to the gRCS of events that have occurred.

The diagram below shows the communication sequence to receive a message status from the robotic vehicle.

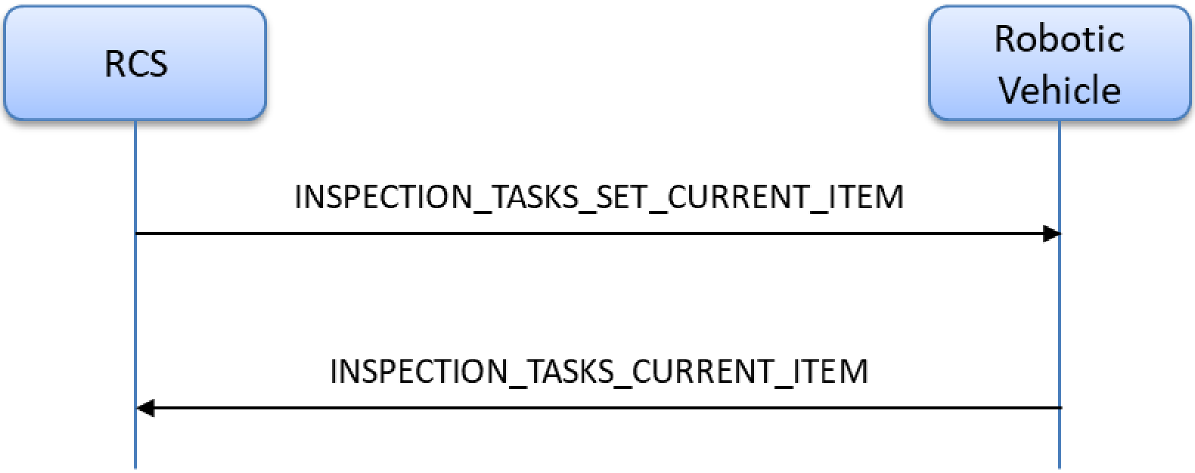


1. When the robotic vehicle detects an important event that may be of interest to the gRCS operator, it sends a TEXT_STATUS message (see section 2.1.26) containing a brief description of the event that has occurred.
    a. These events contain an identifier to indicate the severity of the event that has occurred (see section 2.2.8), either to indicate that it is a warning or an error, or it is simply an informational event for the operator.
2. The gRCS receives this message and processes it, without the need to send a confirmation of its receipt.

1.10 Set current inspection task item to the Robotic Vehicle

Once the robotic vehicle has received the inspection task list and the inspection has started, the gRCS can command the robotic vehicle the specific task to perform.

The diagram below shows the communication sequence to set the current inspection task item to the robotic vehicle. This diagram assumes that all operations are performed successfully.


RCS
Robotic Vehicle
INSPECTION_TASKS_SET_CURRENT_ITEM
INSPECTION_TASKS_CURRENT_ITEM

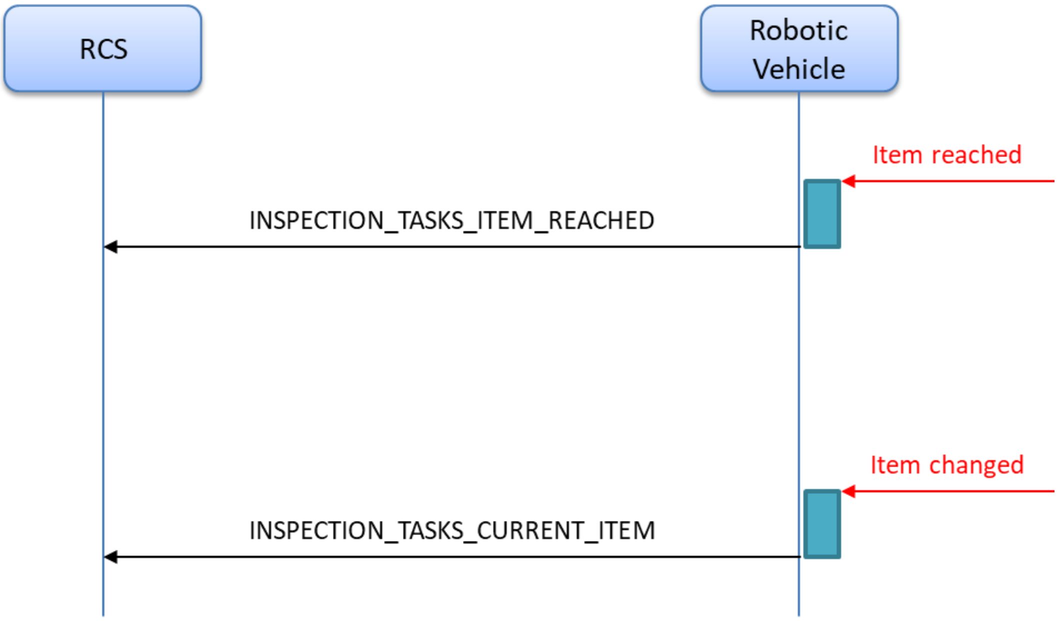
In more detail, the sequence of operations is:

1. The gRCS sends INSPECTION_TASKS_SET_CURRENT_ITEM message (see section 2.1.27), specifying the new sequence number (seq).
    a. There is no specific timeout on the INSPECTION_TASKS_SET_CURRENT_ITEM message
2. The robotic vehicle receives message and attempts to update the current inspection task sequence number.
    a. On success, the robotic vehicle must broadcast an INSPECTION_TASKS_CURRENT_ITEM message (see section 2.1.28) containing the current sequence number (seq).
    b. On failure, the robotic vehicle must broadcast a TEXT_STATUS stating the problem.

1.11 Receive inspection tasks progress from the Robotic Vehicle

During the execution of the mission, the robotic vehicle sends messages with the progress of the inspection tasks.

The diagram below shows the communication sequence to receive the inspection tasks progress from the robotic vehicle.



In more detail, the sequence of operations is:

1. The robotic vehicle must send an INSPECTION_TASKS_ITEM_REACHED message (see section 2.1.29) whenever a new inspection task item is reached. The message contains the seq number of the inspection task item reached.
2. The robotic vehicle must also send an INSPECTION_TASKS_CURRENT_ITEM message if the current inspection task item is changed.

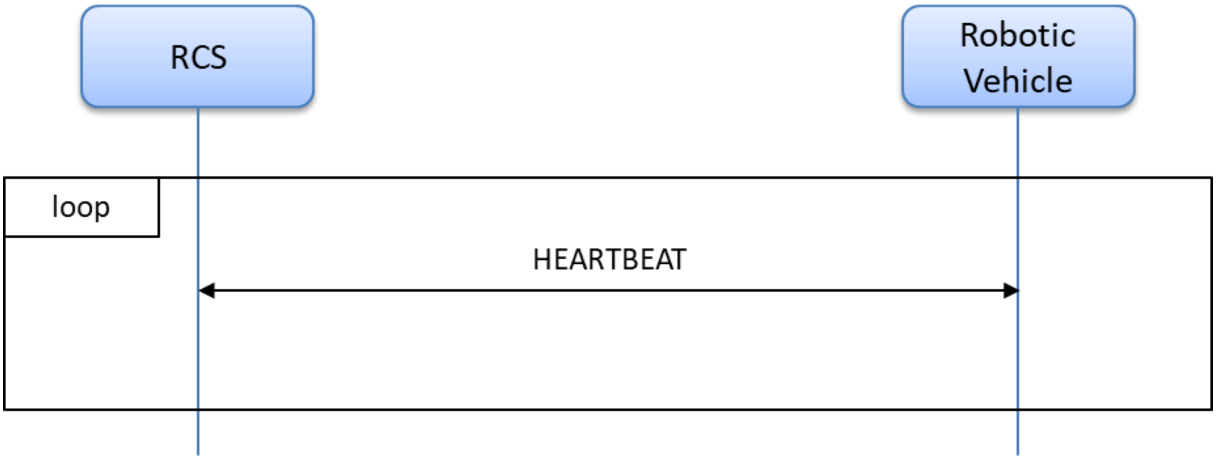
3. The gRCS receives these messages and processes them, without the need to send a confirmation of their receipt.

1.12 Heartbeat protocol

The heartbeat protocol is used to advertise the existence of a system on the MAVLink network.

Systems must regularly send a HEARTBEAT (see section 2.1.30) message and monitor for heartbeats from other systems.

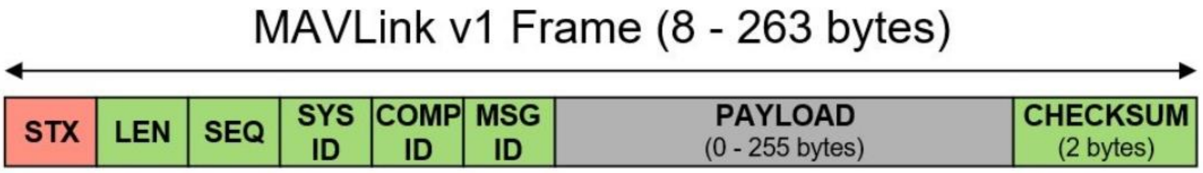
The heartbeat allows other components to discover systems that are connected to the network and infer when they have disconnected. A component is considered to be connected to the network if its HEARTBEAT message is regularly received, and disconnected if a number of expected messages are not received.



2. Data model

2.1 Messages

MAVLink messages are encapsulated in a packet in the following format.



- **Packet start marker:** Protocol-specific start-of-text (STX) marker used to indicate the beginning of a new packet. Any system that does not understand protocol version will skip the packet.
- **Payload length:** Indicates length of the following PAYLOAD section (fixed for a particular message).
- **Packet sequence number:** Used to detect packet loss. Components increment value for each message sent.
- **System ID:** Identifier of system (vehicle) sending the message. Used to differentiate systems on network.
- **Component ID:** Identifier of component sending the message. Used to differentiate components in a system (e.g. autopilot and a camera).
- **Message ID:** Identifier of message type in payload. Used to decode data back into message object.
- **Payload data:** Message data. Content depends on message type.
- **Checksum:** X.25 CRC for message (excluding magic byte).

The following table shows the reserved system identifiers (**System ID**) for each partner.

Partner	System IDs
CATEC	From 30 to 49
ETHZ	From 50 to 69
GEIR	From 70 to 89
RBNK	From 90 to 109
SINTEF	From 110 to 129
USE	From 130 to 149

The messages used at the interface between the gRCS and the robotic vehicle are described below.

2.1.1 INSPECTION_TASKS_COUNT

Send the number of items in the inspection tasks. This is used to initiate inspection tasks upload or as a response to an INSPECTION_TASKS_REQUEST message when downloading the inspection tasks.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
mission_id	uint16_t	Mission identifier
count	uint16_t	Number of inspection tasks items in the sequence

2.1.2 INSPECTION_TASKS_READ

Request inspection tasks item data for a specific sequence number be sent by the recipient using an INSPECTION_TASKS_ITEM message. Used for inspection tasks upload and download.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
seq	uint16_t	Waypoint ID (sequence number). Starts at zero

2.1.3 INSPECTION_TASKS_ITEM

Message encoding an inspection tasks item. Used for inspection tasks upload and download.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
seq	uint16_t	Waypoint ID (sequence number). Starts at zero. Increases monotonically for each waypoint, no gaps in the sequence (0,1,2,3,4)

command	uint16_t	The scheduled action for the waypoint (see MAV_CMD enum)
autocontinue	uint8_t	Autocontinue to next waypoint
param1	float	PARAM1 (see MAV_CMD enum)
param2	float	PARAM2 (see MAV_CMD enum)
param3	float	PARAM3 (see MAV_CMD enum)
param4	float	PARAM4 (see MAV_CMD enum)
x	float	PARAM5: x position in meters
y	float	PARAM6: y position in meters
z	float	PARAM7: z position in meters

2.1.4 INSPECTION_TASKS_ACK

Acknowledgment message when a system completes an inspection tasks operation (e.g. sent by robotic vehicle after it has uploaded all inspection tasks items). The message includes an INSPECTION_TASKS_RESULT indicating either success or the type of failure.

Field Name	**Type**	**Description**
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
type	uint8_t	Inspection tasks operation result (see INSPECTION_TASKS_RESULT enum)

2.1.5 INSPECTION_TASKS_REQUEST

Initiate the inspection tasks download operation from a system.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID

2.1.6 CHECK_LIST_REQUEST

Initiate the checklist download operation from a system.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID

2.1.7 CHECK_LIST_COUNT

Send the number of items in a checklist. This is used as a response to a CHECK_LIST_REQUEST message when downloading the checklist.

Field Name	**Type**	**Description**
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
count	uint16_t	Number of check items in the sequence

2.1.8 CHECK_LIST_READ

Request check item value for a specific sequence number be sent by the recipient using a CHECK_LIST_ITEM message. Used for checklist download.

Field Name	**Type**	**Description**
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
index	uint16_t	Check index. Starts at zero

2.1.9 CHECK_LIST_ITEM

The value of a check item. This is used as a response to CHECK_LIST_READ when downloading the checklist.

Field Name	**Type**	**Description**
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
index	uint16_t	Check index. Starts at zero
name	char[20]	Check name string
description	char[231]	Check description string

2.1.10 CHECK_LIST_ACK

Acknowledgment message when a system completes the checklist download operation. The message includes a CHECK_LIST_RESULT indicating either success or the type of failure.

Field Name	**Type**	**Description**
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
type	uint8_t	Checklist operation result (see CHECK_LIST_RESULT enum)

2.1.11 ALARM_LIST_REQUEST

Initiate the alarm list download operation from a system.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID

2.1.12 ALARM_LIST_COUNT

Send the number of items in an alarm list. This is used as a response to an ALARM_LIST_REQUEST message when downloading the alarm list.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
count	uint16_t	Number of alarm items in the sequence

2.1.13 ALARM_LIST_READ

Request alarm item value for a specific sequence number be sent by the recipient using an ALARM_LIST_ITEM message. Used for alarm list download.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
index	uint16_t	Alarm index. Starts at zero

2.1.14 ALARM_LIST_ITEM

The value of an alarm item. This is used as a response to an ALARM_LIST_READ message when downloading the alarm list.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
index	uint16_t	Alarm index. Starts at zero
name	char[20]	Alarm name string
description	char[231]	Alarm description string

2.1.15 ALARM_LIST_ACK

Acknowledgment message when a system completes the alarm list download operation. The message includes an ALARM_LIST_RESULT indicating either success or the type of failure.

Field Name	**Type**	**Description**
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
type	uint8_t	Alarm list operation result (see ALARM_LIST_RESULT enum)

2.1.16 HL_ACTION_LIST_REQUEST

Initiate the high-level action list download operation from a system.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID

2.1.17 HL_ACTION_LIST_COUNT

Send the number of items in a high-level action list. This is used as a response to a HL_ACTION_LIST_REQUEST message when downloading the high-level action list.

Field Name	**Type**	**Description**
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
count	uint16_t	Number of high-level action items in the sequence

2.1.18 HL_ACTION_LIST_READ

Request high-level action item value for a specific sequence number be sent by the recipient using an HL_ACTION_LIST_ITEM message. Used for high-level action list download.

Field Name	**Type**	**Description**
target_system	uint8_t	System ID

target_component	uint8_t	Component ID
index	uint16_t	High-level action index. Starts at zero

2.1.19 HL_ACTION_LIST_ITEM

The value of a high-level action item. This is used as a response to an HL_ACTION_LIST_READ message when downloading the high-level action list.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
index	uint16_t	High-level action index. Starts at zero
command	uint16_t	Command ID of the command associated with the high-level action (see MAV_CMD enum)
name	char[20]	High-level action name string
description	char[229]	High-level action description string

2.1.20 HL_ACTION_LIST_ACK

Acknowledgment message when a system completes the high-level action list download operation. The message includes a HL_ACTION_LIST_RESULT indicating either success or the type of failure.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
type	uint8_t	High-level action list operation result (see HL_ACTION_LIST_RESULT enum)

2.1.21 COMMAND_LONG

Message to encode a command to be sent to a specific system. The message encodes commands into up to 7 float parameters.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
command	uint16_t	Command ID of command to send (see MAV_CMD enum)
confirmation	uint8_t	Number of resends: • 0: First transmission • 1-255: Confirmation transmissions
param1	float	Parameter 1 for the specific command
param2	float	Parameter 2 for the specific command
param3	float	Parameter 3 for the specific command
param4	float	Parameter 4 for the specific command
param5	float	Parameter 5 for the specific command
param6	float	Parameter 6 for the specific command
param7	float	Parameter 7 for the specific command

2.1.22 COMMAND_ACK

Command acknowledgement. Includes result (success, failure, still in progress) and may include progress information and additional detail about failure reasons.

Report status of a command. Includes feedback whether the command was executed.

Field Name	Type	Description
command	uint16_t	Command ID of acknowledged command (see MAV_CMD enum)
result	uint8_t	Result of command (see MAV_RESULT enum)

2.1.23 LOCAL_POSITION_NED

The local position of a specific system. Coordinate frame is right-handed, Z-axis down (aeronautical frame, NED / north-east-down convention).

Field Name	Type	Description
time_boot_ms	uint32_t	Timestamp (time since system boot), in milliseconds
x	float	X Position, in meters
y	float	Y Position, in meters
z	float	Z Position, in meters
vx	float	X Speed, in meters/seconds
vy	float	Y Speed, in meters/seconds
vz	float	Z Speed, in meters/seconds

2.1.24 ATTITUDE_QUATERNION

The attitude in the aeronautical frame (right-handed, Z-down, X-front, Y-right), expressed as quaternion. Quaternion order is w, x, y, z and a zero rotation would be expressed as (1 0 0 0).

Field Name	Type	Description
time_boot_ms	uint32_t	Timestamp (time since system boot), in milliseconds
q1	float	Quaternion component 1, w (1 in null-rotation)
q2	float	Quaternion component 2, x (0 in null-rotation)
q3	float	Quaternion component 3, y (0 in null-rotation)
q4	float	Quaternion component 4, z (0 in null-rotation)
rollspeed	float	Roll angular speed, in radians/seconds
pitchspeed	float	Pitch angular speed, in radians/seconds
yawspeed	float	Yaw angular speed, in radians/seconds

2.1.25 ALARM_STATUS

This message is used to inform the gRCS of the robotic vehicle's alarm status (indicated by its index).

Field Name	Type	Description
time_boot_ms	uint32_t	Timestamp (time since system boot), in milliseconds
index	uint16_t	Alarm index
status	uint8_t	Alarm status (see ALARM_STATUS_FLAGS enum)
errors_count	uint16_t	Alarm errors count since system boot
warns_count	uint16_t	Alarm warnings count since system boot

2.1.26 TEXT_STATUS

This message is used to inform to the gRCS of any type of warning or error that occurred in the robotic vehicle. It can even be used to send information messages to the gRCS.

Field Name	Type	Description
severity	uint8_t	Severity of status (see TEXT_STATUS_FLAGS enum)
text	char[254]	Status text message, without null termination character

2.1.27 INSPECTION_TASKS_SET_CURRENT_ITEM

Set the inspection task item with sequence number seq as current item.

Field Name	Type	Description
target_system	uint8_t	System ID
target_component	uint8_t	Component ID
seq	uint16_t	Waypoint ID (sequence number). Starts at zero

2.1.28 INSPECTION_TASKS_CURRENT_ITEM

Message that announces the sequence number of the current active inspection task item. The robotic vehicle will move towards this item of the inspection tasks.

Field Name	Type	Description
seq	uint16_t	Waypoint ID (sequence number). Starts at zero

2.1.29 INSPECTION_TASKS_ITEM_REACHED

Message that announces that a certain item of the inspection tasks has been reached.

Field Name	Type	Description
seq	uint16_t	Waypoint ID (sequence number). Starts at zero

2.1.30 HEARTBEAT

The heartbeat message shows that a system is present and responding.

https://mavlink.io/en/messages/common.html#HEARTBEAT

2.2 Enumerations

2.2.1 MAV_CMD

Command identifier used in INSPECTION_TASKS_ITEM and COMMAND_LONG messages. For each command identifier, the values of up to 7 parameters that are packaged within messages are defined.

- MAV_CMD_NAV_WAYPOINT

https://mavlink.io/en/messages/common.html#MAV_CMD_NAV_WAYPOINT

- MAV_CMD_NAV_LOITER_UNLIM

https://mavlink.io/en/messages/common.html#MAV_CMD_NAV_LOITER_UNLIM

- MAV_CMD_NAV_LOITER_TIME

https://mavlink.io/en/messages/common.html#MAV_CMD_NAV_LOITER_TIME

- MAV_CMD_NAV_RETURN_TO_LAUNCH

https://mavlink.io/en/messages/common.html#MAV_CMD_NAV_RETURN_TO_LAUNCH

- MAV_CMD_NAV_LAND

https://mavlink.io/en/messages/common.html#MAV_CMD_NAV_LAND

- MAV_CMD_NAV_TAKEOFF

https://mavlink.io/en/messages/common.html#MAV_CMD_NAV_TAKEOFF

- MAV_CMD_DO_JUMP

https://mavlink.io/en/messages/common.html#MAV_CMD_DO_JUMP

- MAV_CMD_DO_CHANGE_SPEED

https://mavlink.io/en/messages/common.html#MAV_CMD_DO_CHANGE_SPEED

- MAV_CMD_DO_CHANGE_ALTITUDE

https://mavlink.io/en/messages/common.html#MAV_CMD_DO_CHANGE_ALTITUDE

- MAV_CMD_DO_SET_HOME

https://mavlink.io/en/messages/common.html#MAV_CMD_DO_SET_HOME

- MAV_CMD_DO_LAND_START

https://mavlink.io/en/messages/common.html#MAV_CMD_DO_LAND_START

- MAV_CMD_DO_PAUSE_CONTINUE

https://mavlink.io/en/messages/common.html#MAV_CMD_DO_PAUSE_CONTINUE

- MAV_CMD_DO_SET_MISSION_CURRENT

https://mavlink.io/en/messages/common.html#MAV_CMD_DO_SET_MISSION_CURRENT

- MAV_CMD_COMPONENT_ARM_DISARM

https://mavlink.io/en/messages/common.html#MAV_CMD_COMPONENT_ARM_DISARM

- MAV_CMD_NAV_WAYPOINT_QUATERNION

Param	Label Name	Description
1	Quaternion W	Quaternion component w (1 in null-rotation)
2	Quaternion X	Quaternion component x (0 in null-rotation)
3	Quaternion Y	Quaternion component y (0 in null-rotation)
4	Quaternion Z	Quaternion component z (0 in null-rotation)
5	Position X	Position component x, in meters
6	Position Y	Position component y, in meters
7	Position Z	Position component z, in meters

- MAV_CMD_DO_SET_HOME_QUATERNION

Param	Label Name	Description
1	Quaternion W	Quaternion component w (1 in null-rotation)
2	Quaternion X	Quaternion component x (0 in null-rotation)
3	Quaternion Y	Quaternion component y (0 in null-rotation)
4	Quaternion Z	Quaternion component z (0 in null-rotation)

5	Position X	Position component x, in meters
6	Position Y	Position component y, in meters
7	Position Z	Position component z, in meters

2.2.2 MAV_RESULT

Result of command operation (in a COMMAND_ACK message).

Value	Field Name	Description
0	MAV_RESULT_ACCEPTED	Command is valid (is supported and has valid parameters) and was executed.
1	MAV_RESULT_TEMPORARILY_REJECTED	Command is valid but cannot be executed at this time. This is used to indicate a problem that should be fixed just by waiting (e.g. a state machine is busy, cannot arm because have not got GPS lock, etc.). Retrying later should work.
2	MAV_RESULT_DENIED	Command is invalid (is supported but has invalid parameters). Retrying same command and parameters will not work.
3	MAV_RESULT_UNSUPPORTED	Command is not supported (unknown).
4	MAV_RESULT_FAILED	Command is valid, but execution has failed. This is used to indicate any non-temporary or unexpected problem, i.e. any problem that must be fixed before the command can succeed/be retried. For example, attempting to write a file when out of memory, attempting to arm when sensors are not calibrated, etc.
5	MAV_RESULT_IN_PROGESS	Command is valid and is being executed. This will be followed by further progress updates, i.e. the component may send further COMMAND_ACK messages with result MAV_RESULT_IN_PROGRESS (at a rate decided by the implementation), and must terminate by sending a COMMAND_ACK message with final result of the operation. The progress field of COMMAND_ACK message can be used to indicate the progress of the operation.

6	MAV_RESULT_CANCELLED	Command has been cancelled (as a result of receiving a COMMAND_CANCEL message).

2.2.3 INSPECTION_TASKS_RESULT

Result of inspection tasks operation (in an INSPECTION_TASKS_ACK message).

Value	Field Name	Description
0	INSPECTION_TASKS_ACCEPTED	Inspection tasks accepted OK.
1	INSPECTION_TASKS_ERROR	Generic error.
2	INSPECTION_TASKS_UNSUPPORTED	Command is not supported.
3	INSPECTION_TASKS_NO_SPACE	Inspection tasks items exceed storage space.
4	INSPECTION_TASKS_INVALID	One of the parameters has an invalid value.
5	INSPECTION_TASKS_INVALID_PARAM1	Param1 has an invalid value.
6	INSPECTION_TASKS_INVALID_PARAM2	Param2 has an invalid value.
7	INSPECTION_TASKS_INVALID_PARAM3	Param3 has an invalid value.
8	INSPECTION_TASKS_INVALID_PARAM4	Param4 has an invalid value.
9	INSPECTION_TASKS_INVALID_PARAM5	Param5 (x) has an invalid value.
10	INSPECTION_TASKS_INVALID_PARAM6	Param6 (y) has an invalid value.
11	INSPECTION_TASKS_INVALID_PARAM7	Param7 (z) has an invalid value.

12	INSPECTION_TASKS_INVALID_SEQUENCE	Inspection tasks item received out of sequence.
13	INSPECTION_TASKS_CANCELLED	Current inspection tasks operation cancelled (inspection tasks upload or download).

2.2.4 CHECK_LIST_RESULT

Result of checklist downloading operation (in a CHECK_LIST_ACK message).

Value	Field Name	Description
0	CHECK_LIST_ACCEPTED	Checklist accepted OK.
1	CHECK_LIST_ERROR	Generic error.
2	CHECK_LIST_NO_SPACE	Checklist items exceed storage space.
3	CHECK_LIST_INVALID_SEQUENCE	Checklist item received out of sequence.
4	CHECK_LIST_CANCELLED	Current checklist download operation cancelled.

2.2.5 ALARM_LIST_RESULT

Result of alarm list downloading operation (in an ALARM_LIST_ACK message).

Value	Field Name	Description
0	ALARM_LIST_ACCEPTED	Alarm list accepted OK.
1	ALARM_LIST_ERROR	Generic error.
2	ALARM_LIST_NO_SPACE	Alarm list items exceed storage space.
3	ALARM_LIST_INVALID_SEQUENCE	Alarm list item received out of sequence.

4	ALARM_LIST_CANCELLED	Current alarm list download operation cancelled.

2.2.6 HL_ACTION_LIST_RESULT

Result of high-level action list downloading operation (in an HL_ACTION_LIST_ACK message).

Value	Field Name	Description
0	HL_ACTION_LIST_ACCEPTED	High-level action list accepted OK.
1	HL_ACTION_LIST_ERROR	Generic error.
2	HL_ACTION_LIST_NO_SPACE	High-level action list items exceed storage space.
3	HL_ACTION_LIST_INVALID_SEQUENCE	High-level action list item received out of sequence.
4	HL_ACTION_LIST_CANCELLED	Current high-level action list download operation cancelled.

2.2.7 ALARM_STATUS_FLAGS

Alarm status flags used in ALARM_STATUS messages.

Value	Field Name	Description
0	ALARM_STATUS_OK	Alarm in ok status
1	ALARM_STATUS_WARNING	Alarm in warning status
2	ALARM_STATUS_ERROR	Alarm in error status

2.2.8 TEXT_STATUS_FLAGS

Indicates the severity level, used for TEXT_STATUS messages to indicate their relative urgency.

Value	Field Name	Description
0	TEXT_STATUS_ERROR	Indicates an error in systems.
1	TEXT_STATUS_WARNING	Indicates about a possible future error if this is not resolved within a given timeframe. Example would be a low battery warning.
2	TEXT_STATUS_INFO	Normal operational messages. Useful for logging. No action is required for these messages.