[QA Team] Survey on runtime verification and field-based testing of robotic systems.

This survey is part of an exploratory study on **guidelines** for runtime verification and field-based testing of robotic systems. We build guidelines taking inspiration from both academics and practitioners, looking into scientific papers and both proof-of-concept implementations and established tools from github. The long-term goal is to build better methods and tools for gathering confidence in robotic systems, which requires proper infrastructure. The guidelines aim to assist both developers and QA teams to perform runtime verification (RV) and field-based testing (FT) in the context of ROS. Now, we need to understand whether the guidelines are aligned with best practices by surveying experts in the domain.

Instructions: The questionnaire consists of three parts.
1st. Familiarize yourself with an overview of the guidelines (Link: http://bit.ly/3FE7vXN).
2nd. Provide your opinion on usefulness, clarity, and applicability of the guidelines. We also ask for suggestions to improve methods and tools for runtime verification and field-based testing of ROS-based robotic systems.
3rd. We briefly ask about your experience and background to put your responses into context.
* For a deeper understanding of our stance on **field-based testing**, check doi: 10.1145/3447240
* For a deeper understanding of our stance on **runtime verification**, check doi: 10.1007/s10009-021-00609-z

The survey takes approximately 15 minutes to complete. We will keep your responses confidential and only store them for analysis. The report will only contain aggregated summaries of the responses and anonymized quotes.

Please let us know if you have any questions. You can contact directly <ricardo.caldas@chalmers.se>.

Thank you in advance for your valuable contribution. Your participation will significantly contribute to advancing the field of testing for robotic systems.

Sincerely,

Ricardo Caldas (Chalmers University of Technology, Sweden)
Juan Antonio Piñera García (Gran Sasso Science Institute, Italy)
Matei Schiopu (Chalmers University of Technology, Sweden)
Genaína Nunes Rodrigues (Universidade de Brasília, Brazil)
Patrizio Pelliccione (Gran Sasso Science Institute, Italy)
Thorsten Berger (Ruhr University Bochum, Germany and Chalmers|University of Gothenburg, Sweden)

* indicates required question

1. Informed Consent *

Tick all that apply.

□ I hereby consent to participate in this experiment. I acknowledge that participation is voluntary and I can leave at any time throughout the experiment.

□ I have read the instruction materials and I have access to the guidelines' overview.

2. Have you ever worked with robotics software? (e.g., development, testing, research) *

Mark only one oval.

○ Yes

○ No

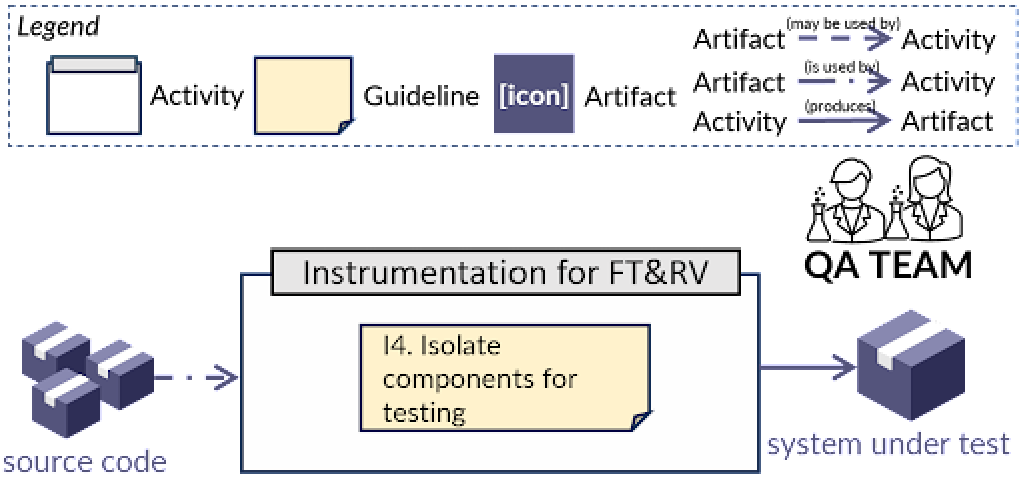
Instrumentation

This section queries about the guidelines on instrumentation for runtime verification and field-based testing.

Definition: With the source code in hands, either developers or the QA team may perform code instrumentation, which consists of modifying the source code to enabling interactions between internal variables with monitors.

Please analyze the graphical representation of the instrumentation activity and respond to the following questions.

Instrumentation activity and guidelines


Legend
Activity
Guideline
[icon] Artifact
Artifact (may be used by) Activity
Artifact (is used by) Activity
Activity (produces) Artifact
QA TEAM
Instrumentation for FT&RV
I4. Isolate components for testing
source code
system under test

3. **Guideline I4.** *Isolation of components is an important feature to enable field-based testing (by following the ``let it crash" philosophy introduced by Netflix web:chaos-monkey) while avoiding the crash of the system. Isolation permits catching immediately the component that is crashing and executing countermeasures to keep the system up. Examples of solutions for isolation are: (i) the use of proxy nodes of ROSRV (git: cansuerdogan/ROSRV), or (ii) introducing intermediary nodes between original nodes by exploiting the topic remapping functionality (wiki: remap) of ROSMonitoring (git: autonomy-and-verification-uol/ROSMonitoring), which enables swapping topic names just before running the application.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

4. Do we miss any guideline or do you have any suggestion for improvement? Please elaborate.

Specifying (un)desired behavior

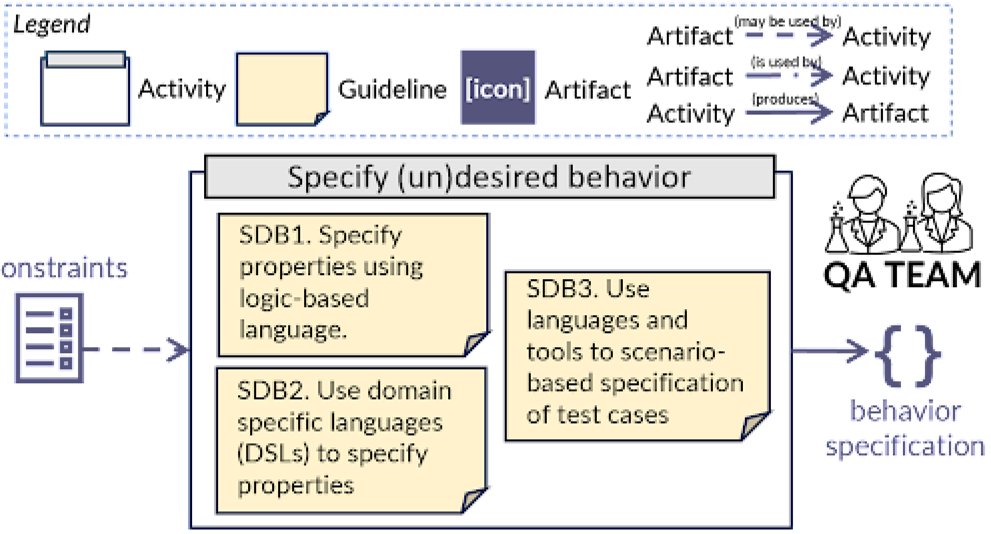
This section queries about the guidelines on specification of (un)desired behavior for runtime verification and field-based testing

.

Definition: Specifying (un)desired behavior asks for the means to abstracting the system behavior in terms of states and events (aka state changes) and how specification languages can be used to describe properties of a set of such states or events.

Please analyze the graphical representation of the specifying (un)desired behavior activity and respond to the following questions.

Specifying (un)desired behavior activity and guidelines


Legend
Activity
Guideline
[icon] Artifact
Artifact (may be used by) Activity
Artifact (is used by) Activity
Activity (produces) Artifact
Specify (un)desired behavior
SDB1. Specify properties using logic-based language.
SDB2. Use domain specific languages (DSLs) to specify properties
SDB3. Use languages and tools to scenario-based specification of test cases
constraints
QA TEAM
behavior specification

5. **SDB1.** *The QA team should be prepared to specify properties using unambiguous and precise languages like logic-based languages, as often required by verification tools. User-friendly instruments, like specification patterns (web: ps-patterns), might facilitate the error-prone specification process and make the specification accessible to people lacking expertise in logic. For example, HAROS (git: git-afsantos/haros) uses a logic-based language called HPL for property specification that is used to synthesize runtime monitors for testing and verification.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

6. **SDB2**. *In complement to logic-based instruments, the QA team may opt to use verification tools that allow code-like specifications of properties to simplify the definition of the desired behavior. For example, ROSMonitoring (git: autonomy-and-verification-uol/ROSMonitoring) allows for code-like specifications of properties in a domain specific language (DSL) targeted to the properties such as writing assertions over the robot's position using if-else constructs.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

7. **SDB3**. *The QA team might integrate scenario specification languages and tools to enable a systematic exploration of real-world situations and conditions for ROS-based applications. For example, Geoscenario (git: rodrigoqueiroz/geoscenarioserver) uses behavior trees to program dynamic interactions between the system-under-test and other vehicles in the scenario. In addition, SCENIC (git: BerkeleyLearnVerify/Scenic) is a probability-based programming language that enables the specification of rare events in environment models that are used to generate test cases for vehicles running on the CARLA simulator (git: carla-simulator/carla) that may be further integrated to testing ROS-based systems.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

8. Do we miss any guideline or do you have any suggestion for improvement? Please elaborate.

Generate monitors and test cases

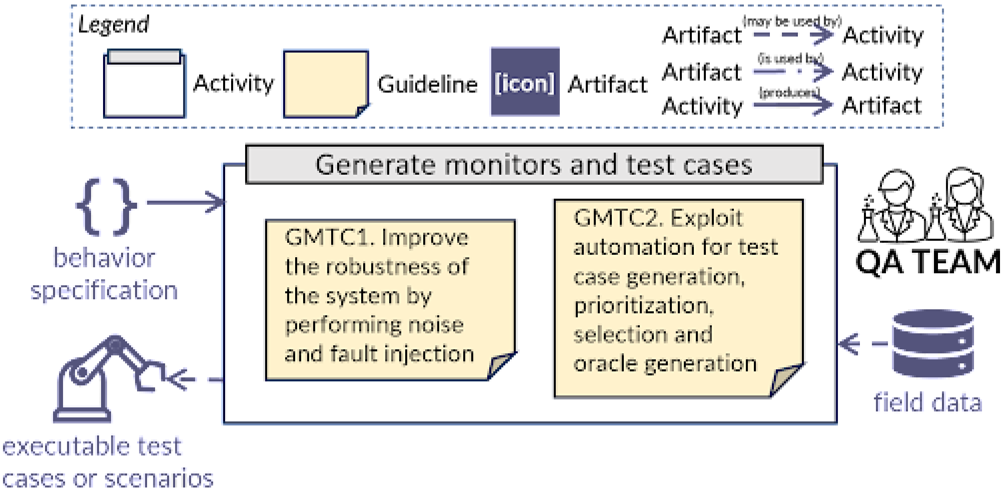
This section queries about the guidelines on generating monitors and test cases for runtime verification and field-based testing

.

Definition: Generate monitors and test cases synthesizes either test cases (encoding an oracle) or test scenarios that may accept or reject an observed execution trace.

Please analyze the graphical representation of the generating monitors and test cases activity and respond to the following questions.

Generate monitors and test cases activity and guidelines


Legend
Activity
Guideline
[icon] Artifact
Artifact (may be used by) Activity
Artifact (is used by) Activity
Activity (produces) Artifact
Generate monitors and test cases
GMTC1. Improve the robustness of the system by performing noise and fault injection
GMTC2. Exploit automation for test case generation, prioritization, selection and oracle generation
behavior specification
executable test cases or scenarios
QA TEAM
field data

9. **GMTC1**. *The QA team can use tools that generate noise or inject faults to gain confidence that a robotic system will behave safely when faced with unexpected situations. For instance, RoboFuzz (git: sslab-gatech/RoboFuzz) enables the generation of faults (in ROS 2 applications) through message mutation with three intents: violation of physical laws, violation of user-specified properties, and cyber-physical discrepancies. Moreover, RosPenTo (git: jr-robotics/ROSPenTo) enables security assessment by (un-) registering publishers or subscribers, isolating nodes or services, and injecting false data in the messages in ROS 1 applications.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

10. **GMTC2**. *The QA team should exploit automation tools for test case generation, test case selection and oracle generation, as well as other testing activities, to efficiently gain confidence in ROS-based systems in the field. For example, Mithra (pdf: AfzalMithra.pdf) learns oracles from logs generated during the execution, it is motivated by a case from ArduPilot and tested in autonomous racing cars built on ROS. In addition, HAROS (git: git-afsantos/haros) promotes test case generation from properties using a tool called Hypothesis (git: HypothesisWorks/hypothesis). Finally, a technical report (pdf: 2022-iros-roboticstesting.pdf) details how a company building mobile robots for disinfection uses equivalence partitioning for test case selection for the field.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

11. Do we miss any guideline or do you have any suggestion for improvement? Please elaborate.

Prepare execution environment for FT&RV

This section queries about the guidelines on preparing the execution environment for runtime verification and field-based testing

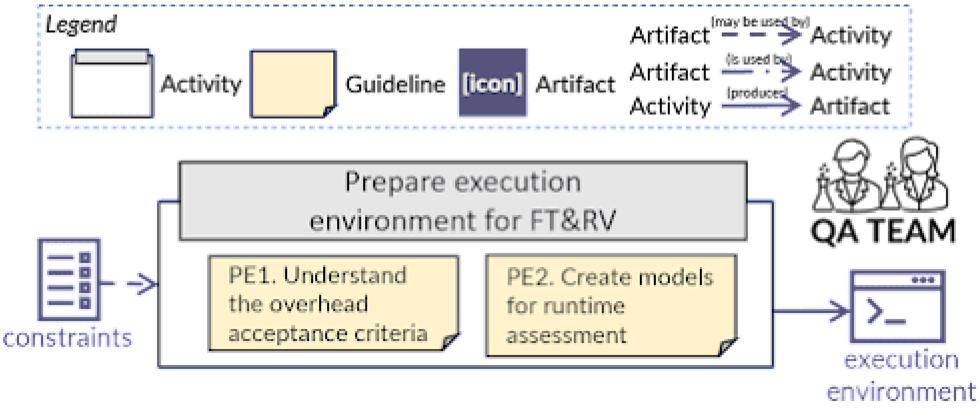
.

Definition: The QA team prepares the execution environment by setting up supporting devices such as stubs and mocks to field-based testing and runtime verification.

Please analyze the graphical representation of the preparing execution environment for runtime verification and field-based testing

activity and respond to the following questions.

Prepare execution environment for FT&RV activity and guidelines


Legend
Activity
Guideline
[icon] Artifact
Artifact [may be used by] Activity
Artifact [is used by] Activity
Activity [produces] Artifact
Prepare execution environment for FT&RV
PE1. Understand the overhead acceptance criteria
PE2. Create models for runtime assessment
QA TEAM
constraints
execution environment

12. **PE1.** *The use of runtime verification or field-based techniques might add computation overhead. The QA team should understand how much overhead is acceptable; this is important to decide on a test strategy that will not severely impact the performance of the running system. Such overhead may be due to monitoring with ros2_tracing (git: ros2/ros2_tracing), component isolation, or security and privacy maintenance overhead with ROSploit (git: seanrivera/rosploit).* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

13. **PE2.** *The QA team might create and exploit models of the system and/or of its environment (a sort of digital twin) for runtime assessment, predictive maintenance, checking alternatives, and so on. For example, they can create a digital twin of the system by using CPSAML (me-big-tuwien-ac-at/cpsaml), or formal tools such as UPPAAL combined with UPPAALTron (doi: 10.1109/ECMR.2015.7324210). In addition, the QA team may use ROS metamodels (ipa-nhg/ros-model) to facilitate the use of tools and graphical plug-ins for reverse engineering models from ROS code.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

14. Do we miss any guideline or do you have any suggestion for improvement? Please elaborate.

_____

_____

_____

_____

_____

System execution for FT&RV

This section queries about the guidelines on preparing the system execution for runtime verification and field-based testing

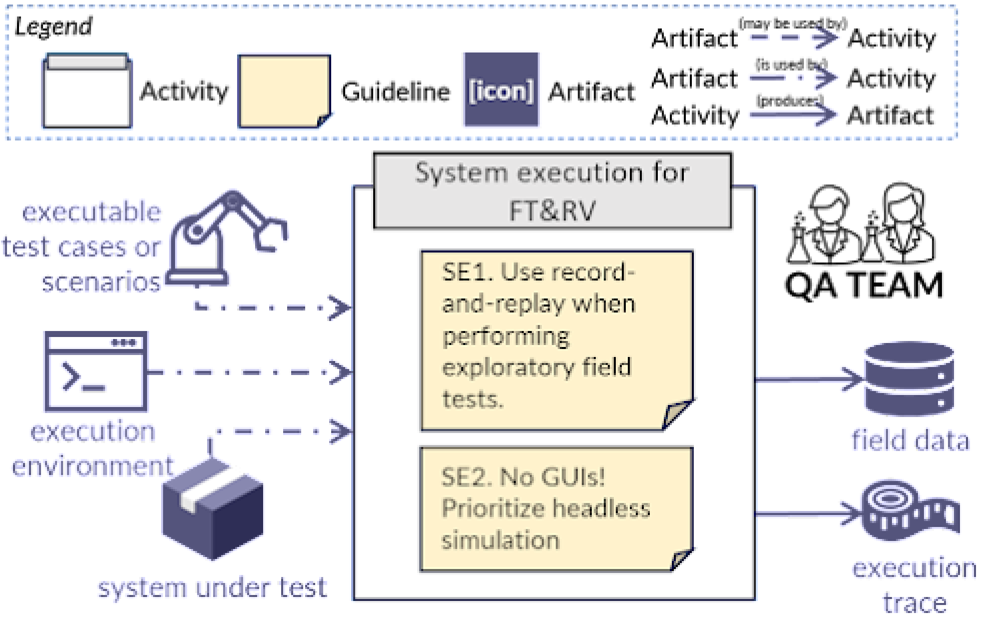
.

Definition: System execution is the activity of running one ore more test cases or test scenarios in a given execution environment to exercise a system under test, which results in an execution trace and field data.

Please analyze the graphical representation of the system execution for runtime verification and field-based testing

activity and respond to the following questions.

System execution for FT&RV activity and guidelines


Legend
Activity
Guideline
[icon]
Artifact
Artifact (may be used by) Activity
Artifact (is used by) Activity
Activity (produces) Artifact
System execution for FT&RV
SE1. Use record-and-replay when performing exploratory field tests.
SE2. No GUIs! Prioritize headless simulation
executable test cases or scenarios
execution environment
system under test
QA TEAM
field data
execution trace

15. **SE1.** *When running exploratory field testing, the QA team should use record-and-playback in order to keep track of the explored field scenarios, simplify error analysis, find and reproduce corner cases, and help with parameter tuning. The standard tool for record-and-playback in ROS is rosbag (wiki: rosbag) but there are a few tool derivations supporting effective record-and-playback. For example, Rerun.io (git: rerun-io/rerun) promotes a graphical interface with a focus on the visualization of bag data leveraging common datatypes used on perception algorithms. In addition, NuBots (git: NUbots/NUbots) uses a genetic algorithm for tuning parameters for the RoboCup over data collected in field explorations after data bags.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

16. **SE2.** *When it is possible to test or verify without human supervision, the QA team should prioritize headless simulation to avoid unnecessary overhead, enable large-scale experimentation, and facilitate integration with CI/CD pipelines. Example of tools supporting headless simulation are Gazebo (git: gazebosim/gz-sim), V-REP (wiki: vrep_ros_bridge), ARGOS (web: argos-sim), MORSE (web: morse-simulator), and MVSim (git: MRPT/mvsim). As a complement, OpenDaVINCI (git: se-research/OpenDaVINCI) interfaces with ROS and has been widely used for testing autonomous driving systems.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

17. Do we miss any guideline or do you have any suggestion for improvement? Please elaborate.

Analysis and Reporting for FT&RV

This section queries about the guidelines on preparing the system execution for runtime verification and field-based testing

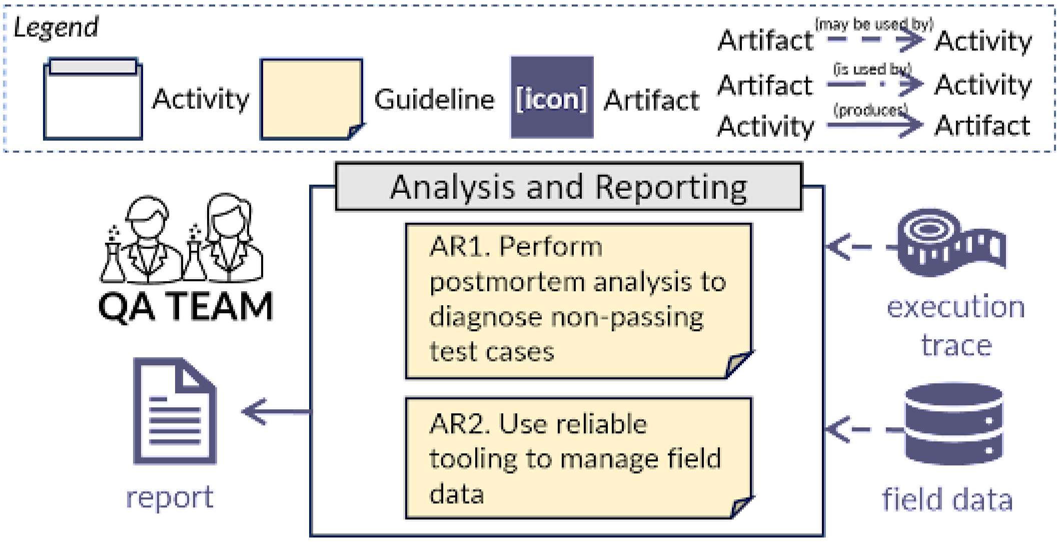
.

Definition:
Analysis and reporting gathers the field data and execution traces generated during test execution and derives evidence, conclusions and finally a report on the observed behavior.

Please analyze the graphical representation of the system execution for runtime verification and field-based testing

activity and respond to the following questions.

Analysis reporting for FT&RV activity and guidelines


Legend
Activity
Guideline
[icon] Artifact
Artifact (may be used by) Activity
Artifact (is used by) Activity
Activity (produces) Artifact
Analysis and Reporting
AR1. Perform postmortem analysis to diagnose non-passing test cases
AR2. Use reliable tooling to manage field data
QA TEAM
report
execution trace
field data

18. **AR1**. *The QA team should perform postmortem analysis and diagnose of non-passing test cases to explain the failures to developers or refine the arguments and confidence in the robotic system. For example, ROS projects may use a combination of Nagios (web: nagios) and ros/diagnostics for monitoring, collecting and aggregating runtime data to diagnose failures. Moreover, CARE (git: softsys4ai/care) may be used for semi-automatic diagnosis of launch file misconfigurations or may rely on approaches such as Rason (git: lsa-pucrs/rason/) for multi-robot diagnosis.* *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

19. **AR2**. The QA team should use reliable tools for field data management to avoid problems with corrupted, unreliable, and/or incomplete data. For example, the warehouse_ros package offers both MongoDB (git: ros-planning/warehouse_ros_mongo) and SQLite (git: ros-planning/warehouse_ros_sqlite) database backend for recording states, scenes, and messages. In addition, the Field Test Tool (git: fkie/field_test_tool) uses the PostgreSQL database manager extended with PostGIS for geolocalization. *

To which extent do you agree that this guideline is ...?

Mark only one oval per row.

	Fully disagree	Disagree	Agree	Fully agree
Useful.	○	○	○	○
Clearly formulated.	○	○	○	○
Applicable to ROS-based systems that you've worked with.	○	○	○	○

20. Do we miss any guideline or do you have any suggestion for improvement? Please elaborate.

Personal Background

To put your response into context, please tell us briefly about your background.

*Reminder:
We will keep your responses confidential and only store them for analysis. The report will only contain aggregated summaries of the responses, and anonymized quotes.

21. How long have you worked with robotics software? *

Mark only one oval.

○ < 1 year
○ 1 -- 3 years
○ 3 -- 5 years
○ 5 -- 10 years
○ > 10 years

22. How does your work in robotics software relate to ROS (Robot Operating System)? *

Mark only one oval.

○ I have contributed to ROS package(s) (e.g., development, testing)
○ I have worked on robotics applications that use ROS packages.
○ I have worked on robotics applications, but I am unsure whether they use ROS packages.
○ I haven't heard of ROS before.
○ I am part of the ROS development team.
○ Other: ______________________

23. Which type of organizations have you worked with robotics? *

Tick all that apply.

□ Academia
□ Industry
□ Government
□ Independent groups (e.g., Open Robotics, hobbists)
□ Individually
□ Other: ________________________

24. What robotics domains have you worked with? *

Tick all that apply.

□ Service robotics
□ Marine robotics
□ Healthcare robotics (e.g., prosthetics, surgery)
□ Industrial automation
□ Construction
□ General-purpose
□ Other: ________________________

25. To receive the results of the study, please enter your email address.

____

26. Could we contact you in case of follow-up questions?

Mark only one oval.

○ Yes, via my email address above
○ No

Thanks a lot for your time and contribution! Your responses are very valuable to the community. Please click **Submit** to finish the survey.

Note: After **submitting** the questionnaire, please consider answering also the questionnaire that targets
runtime verification and field-based testing of ROS-based systems from the **developers perspective**, if you haven't yet. Link: https://forms.gle/9de4Vn9xusZfxcBp9.