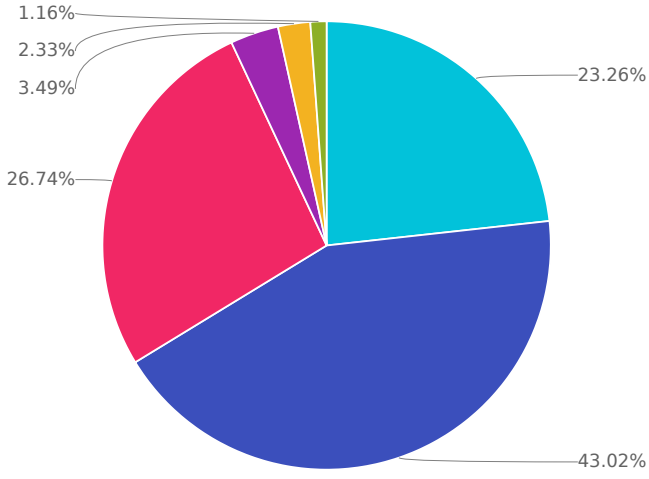
Response Statistics

129	86	65	21
Survey Visits	**Total Responses**	**Completed Responses**	**Partial Responses**

Section 1: Background

Q1

What entity do you work for?



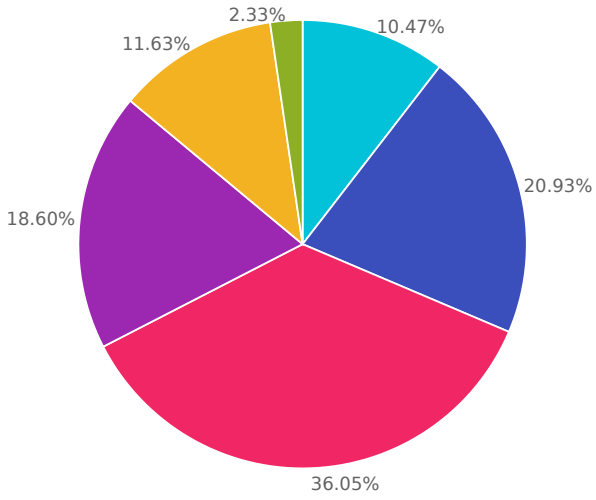
Choices	Response percent	Response count
University or Research Institute	23.26%	20
Small and mid-sized enterprise (less than 250 employees)	43.02%	37
Large corporation (250+ employees)	26.74%	23
Automation or robotic associations or society (nonprofit)	3.49%	3
Self-employed	2.33%	2
Other (Please specify)	1.16%	1

Other (Please specify)

1. Student and Hobbyist

Q2

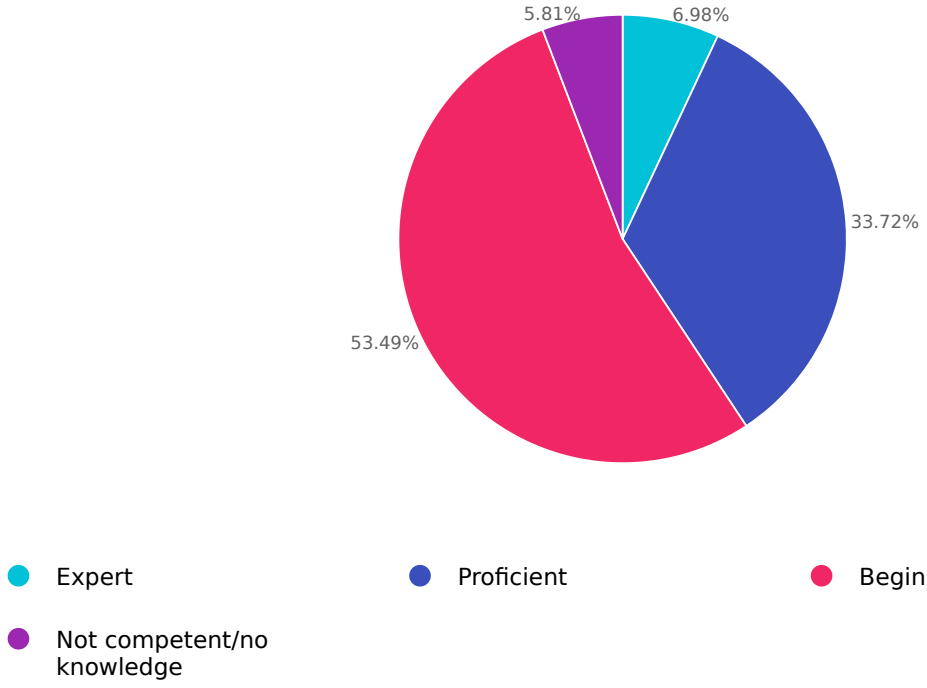
How many years of experience do you have working in the AI and robotics field?



Choices	Response percent	Response count
Less than 1 year	10.47%	9
1-2 years	20.93%	18
3-5 years	36.05%	31
6-10 years	18.60%	16
More than 10 years	11.63%	10
I do not have experience in this field	2.33%	2

Q3

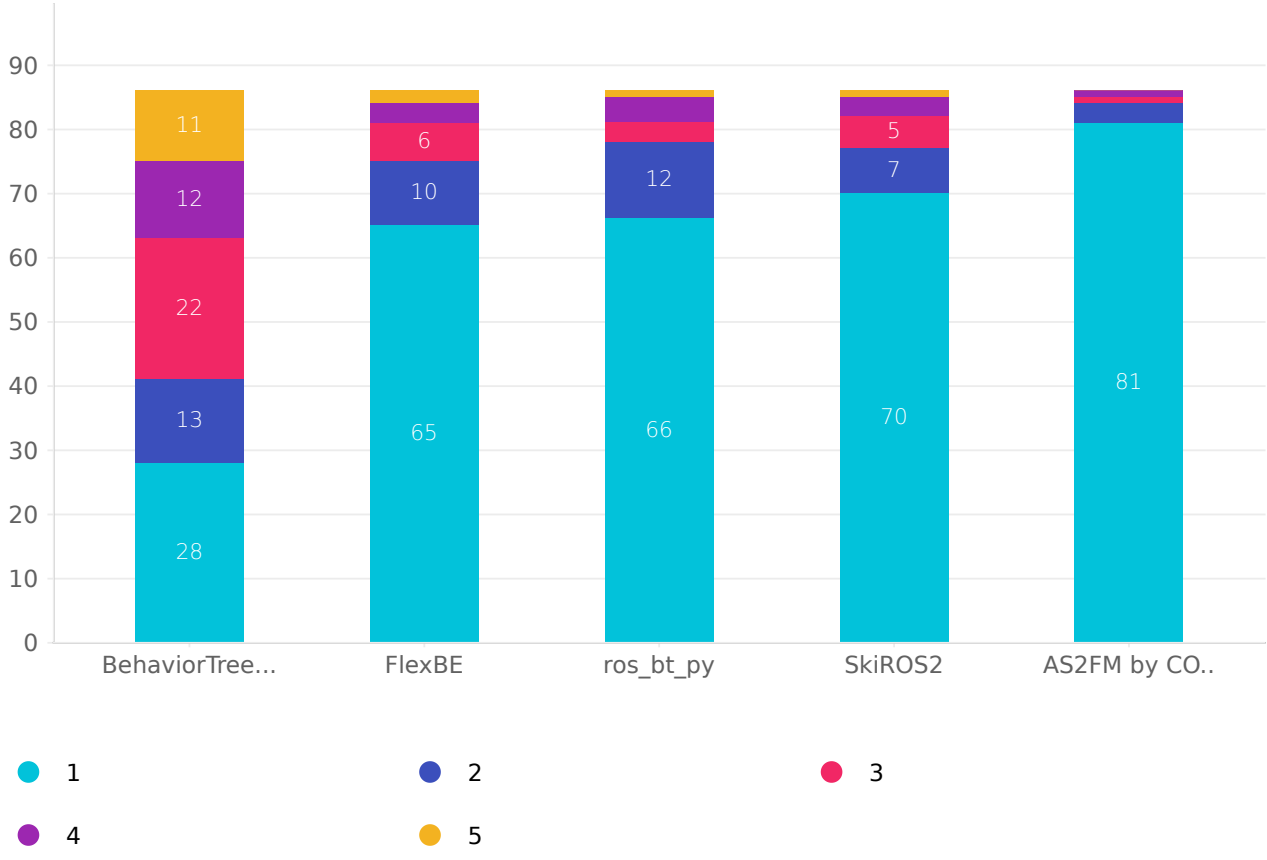
What is your level of understanding in the area of deliberation for autonomous systems (e.g. situation modelling, task planning, skills/capabilities/behavior definition)?



Choices	Response percent	Response count
Expert	6.98%	6
Proficient	33.72%	29
Beginner	53.49%	46
Not competent/no knowledge	5.81%	5

Q4

How familiar were you with each of the following tools before the workshop?



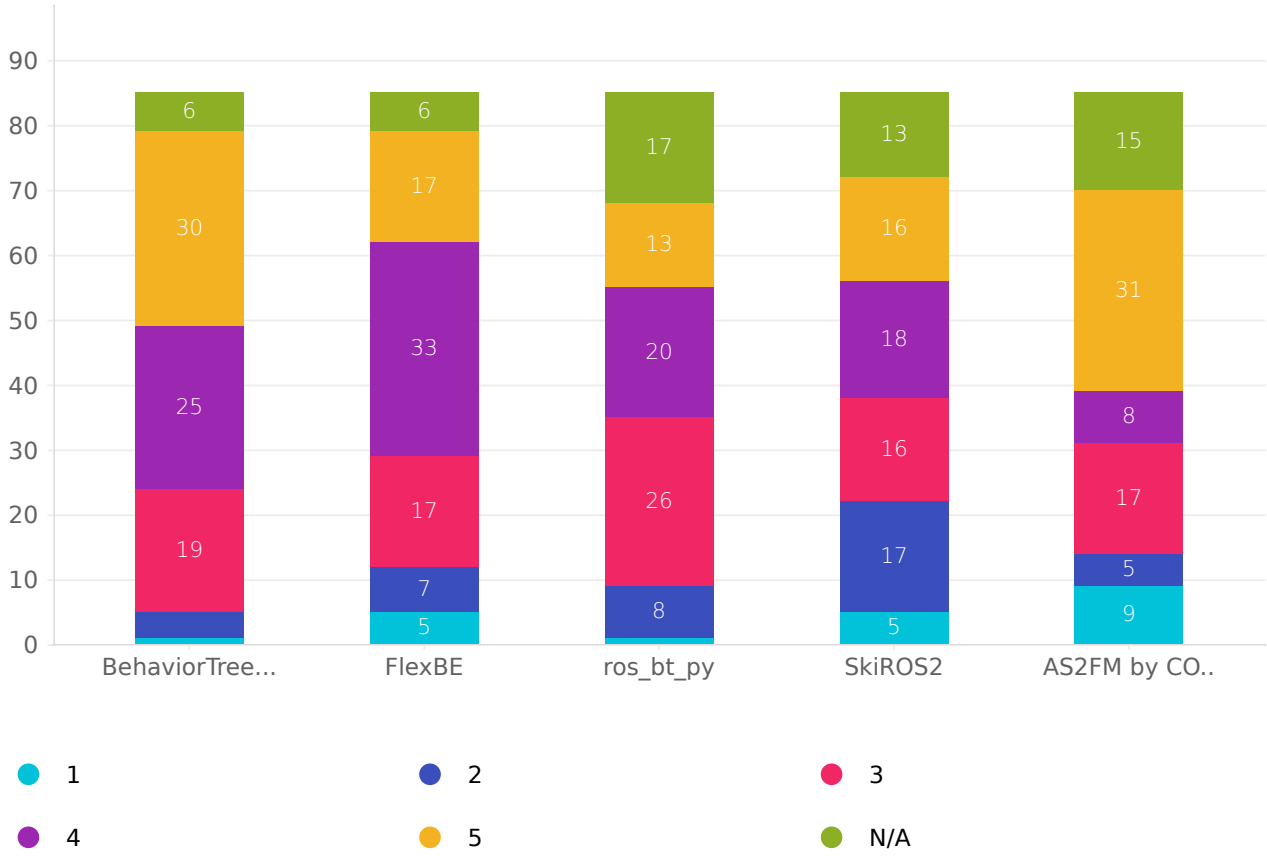
Row	1 (Not at all familiar)	2	3	4	5 (Very familiar)	Average rating	Response count
BehaviorTree.CPP	32.56% (28)	15.12% (13)	25.58% (22)	13.95% (12)	12.79% (11)	2.59	86
FlexBE	75.58% (65)	11.63% (10)	6.98% (6)	3.49% (3)	2.33% (2)	1.45	86
ros_bt_py	76.74% (66)	13.95% (12)	3.49% (3)	4.65% (4)	1.16% (1)	1.40	86
SkiROS2	81.40% (70)	8.14% (7)	5.81% (5)	3.49% (3)	1.16% (1)	1.35	86
AS2FM by CONVINCE	94.19% (81)	3.49% (3)	1.16% (1)	1.16% (1)	0.00% (0)	1.09	86

Average rating: 1.58

Section 2: Workshop Experience

Q5

How easy was it for you to set up and start using each tool in the workshop?

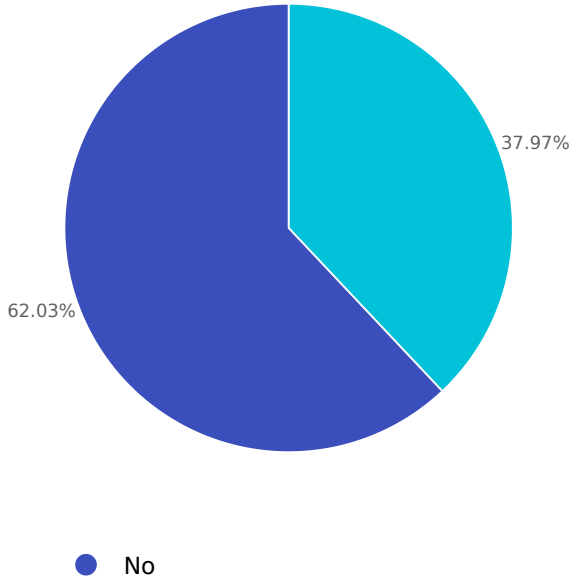


Row	1 (Very difficult)	2	3	4	5 (Very easy)	N/A	Average rating	Response count
BehaviorTree.CPP	1.18% (1)	4.71% (4)	22.35% (19)	29.41% (25)	35.29% (30)	7.06% (6)	4.00	85
FlexBE	5.88% (5)	8.24% (7)	20.00% (17)	38.82% (33)	20.00% (17)	7.06% (6)	3.63	85
ros_bt_py	1.18% (1)	9.41% (8)	30.59% (26)	23.53% (20)	15.29% (13)	20.00% (17)	3.53	85
SkiROS2	5.88% (5)	20.00% (17)	18.82% (16)	21.18% (18)	18.82% (16)	15.29% (13)	3.32	85
AS2FM by CONVINCE	10.59% (9)	5.88% (5)	20.00% (17)	9.41% (8)	36.47% (31)	17.65% (15)	3.67	85

Average rating: 3.64

Q6

Did you encounter any technical issues with the tools at the workshop? *If yes, please specify which tool(s) and describe the issues you experienced.*



Choices	Response percent	Response count
Yes	37.97%	30
No	62.03%	49
Please describe issues you experienced.		31

Please describe issues you experienced.

1. FlexBE: webgui frozen sometimes. Needed to restart it Ros_bt_py: for some of the provided URLs for the webgui, the list of available node was empty
2. The SkiROS2 was for some reason difficult to launch at times. Like I ended up in a state where i had to close down the docker instance and start over before i got the right setup. I'm not entirely sure what went wrong.
3. Skiros2 Gui crash or black screen
4. as2fm_scxml_to_jani main.xml command not found
5. Groot2.appimage was missing a file The binaries for convince were missing. Might be due to pulling the workshop image a week before the conference.
6. ros_bt_py: GUI has some bags when updating nodes. The update button does not work, but when I click in the diagram section and cancel the prompt, it updates it. flexBE or robotsim: the connection to the simulator broke: The robot rendering did not move but the BT kept going
7. Reading PDDLs with SkiROS2 (EOL issue).
8. Docker image was old
9. I had issues with SkiROS2, where i could not load goals from pddl files. However this might be related to me running the docker in WSL, which was not officially supported. Everything else worked fine.
10. Load model into the groot without copying demo

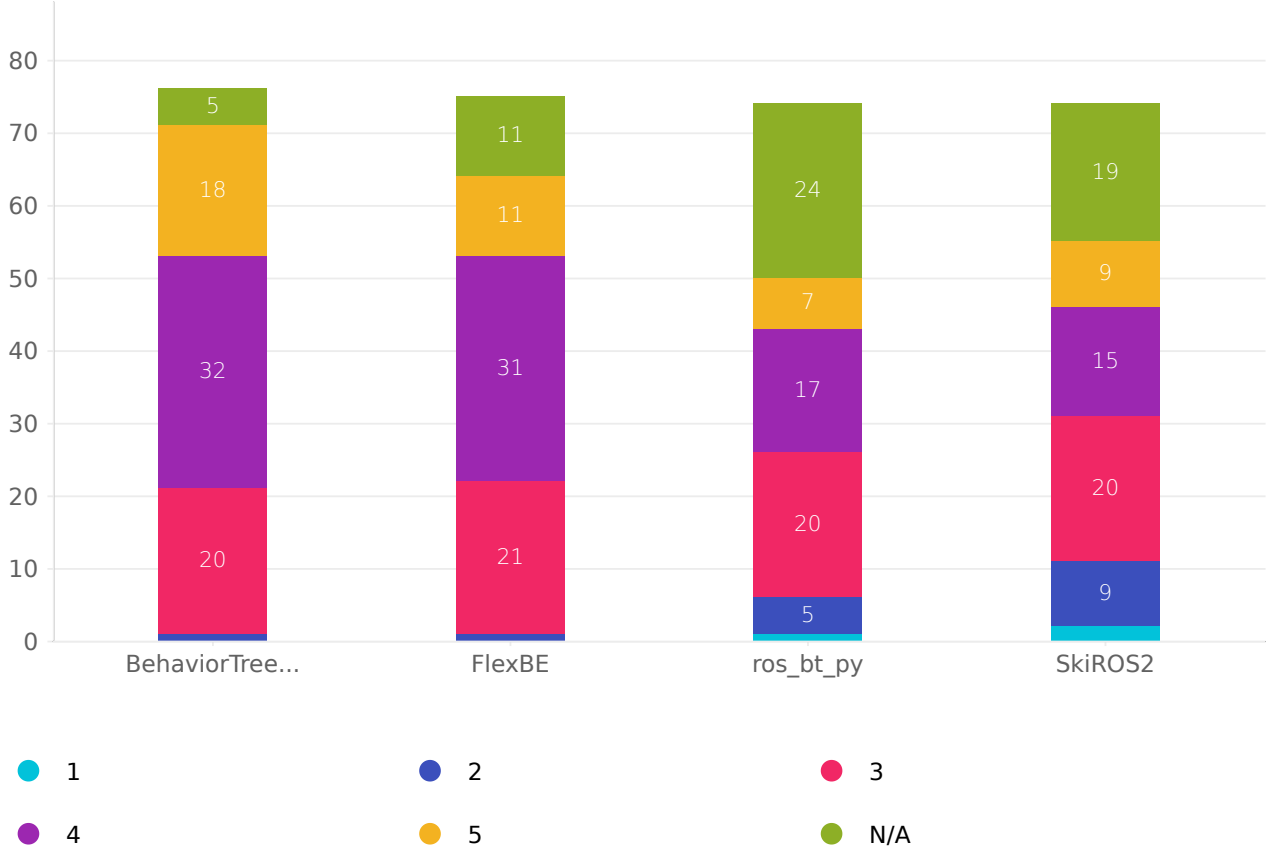
11. Yes, but got a support from FLEXbe team.
12. Had my ros_bt_py ui crash multiple times and had to remake my behaviour tree :)
13. Groot2 does not allow to save a file/model that is incomplete, same with FlexBE GUI. FlexBE is prone to crashing, and you need to restart all ROS processes to get it started again. FlexBE is also overlaying error messages over the actual work area, which is inconvenient.
14. With SkiROS, I was unable to get the live demos to work due to the the workshop getting updated. Ros_bt_py's web client was a little laggy, but otherwise was able to work!
15. With the monitor of FSM it was not adapting to the resolution of my screen, should be enough to add scroll bars to the window
16. SkiROS2, files were updated after I pulled the repository (I did it a few days prior) and the examples didn't run
17. After trying to update the environment, skiros stopped building. I tried to clean and rebuild, pulled the repo and submodules, and even though they were updated, they still didn't work. The only thing that worked for me was to clone the main repo again and rebuilding the image.
18. SkiROS was stuck in trying to plan even after a long while. I simply tried the solution of one of the PDDL files.
19. SkiRos2 did stopped executing unexpectedly during a plan. Probably skill issue.
20. A bit hard to follow the presentation, can slow it down a little. Hard to run the SkiROS pddl-files - gave errors after loading and running. Not so intuitive GUI.
21. I had issues with the ros_bt_py gui. I was not able to update nodes at times, and there was no feedback as to what failure was happening.
22. I needed to start SkiROS2 problem 2 several times until it worked FlexBE died and I lost my progress
23. The survey was closed
24. GUIs crash
25. Forgot to 'git submodule update' and then 'docker compose run -- build base'
26. SkiROS never opened the world view when launched, only the SkiROS gui
27. Great talk!
28. Not enough power plugs for everybody. But just a minor thing.
29. Yes, X-11 / qt plugins not loading.
30. Groot didn't run Smc_storm didn't run
31. I was not able to execute the behaviours for ros_bt_py but I was able to load up the GUI to play around with the system

Section 3: Explore tools functionality

If you feel that any of the answer choices is not entirely relevant to the questions, please use your best judgment to answer or select 'N/A' (Not Applicable).

Q7

How well did each tool allow you to define and implement complex behaviors?

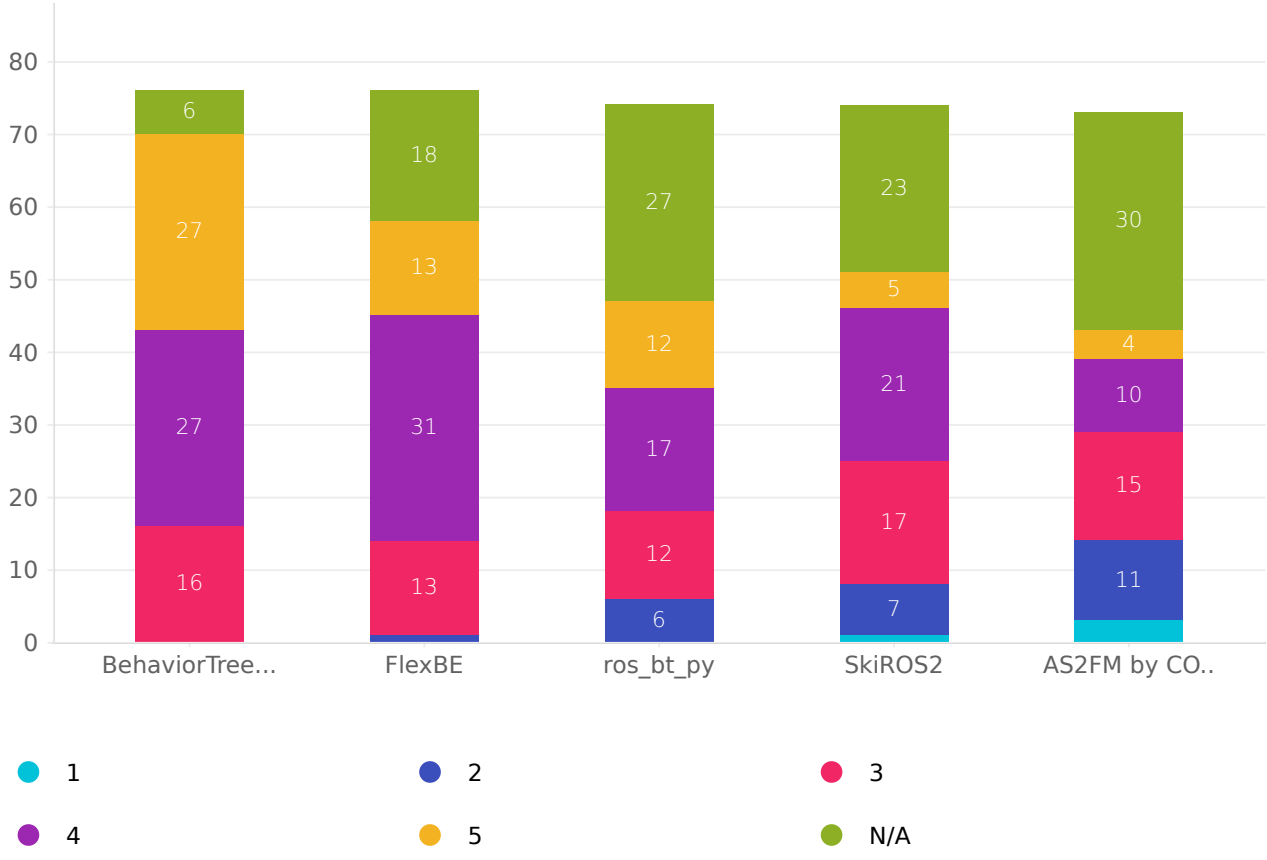


Row	1 (Not well at all)	2	3	4	5 (Very well)	N/A	Average rating	Response count
BehaviorTree.CPP	0.00% (0)	1.32% (1)	26.32% (20)	42.11% (32)	23.68% (18)	6.58% (5)	3.94	76
FlexBE	0.00% (0)	1.33% (1)	28.00% (21)	41.33% (31)	14.67% (11)	14.67% (11)	3.81	75
ros_bt_py	1.35% (1)	6.76% (5)	27.03% (20)	22.97% (17)	9.46% (7)	32.43% (24)	3.48	74
SkiROS2	2.70% (2)	12.16% (9)	27.03% (20)	20.27% (15)	12.16% (9)	25.68% (19)	3.36	74

Average rating: 3.68

Q8

How well did each tool support modularity and reusability of components?

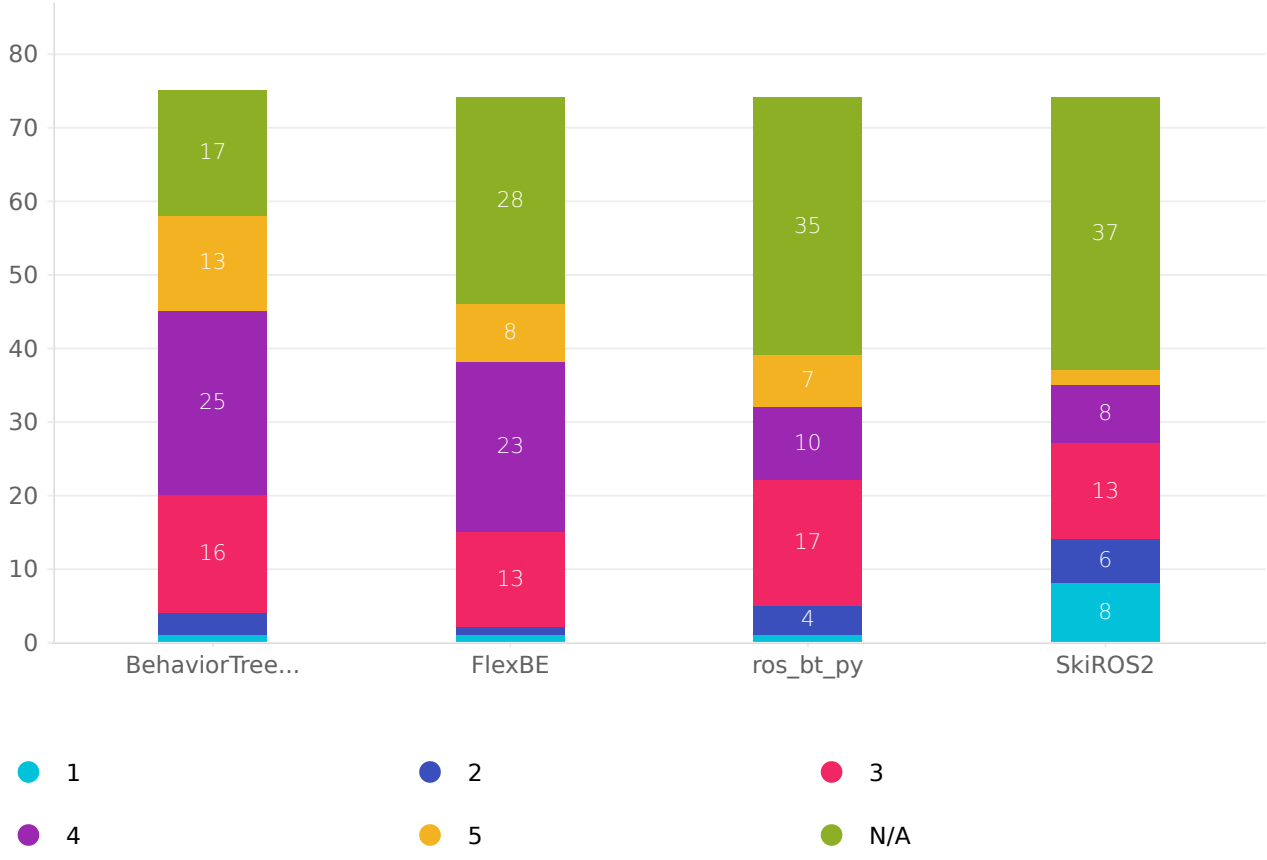


Row	1 (Not well at all)	2	3	4	5 (Very well)	N/A	Average rating	Response count
BehaviorTree.CPP	0.00% (0)	0.00% (0)	21.05% (16)	35.53% (27)	35.53% (27)	7.89% (6)	4.16	76
FlexBE	0.00% (0)	1.32% (1)	17.11% (13)	40.79% (31)	17.11% (13)	23.68% (18)	3.97	76
ros_bt_py	0.00% (0)	8.11% (6)	16.22% (12)	22.97% (17)	16.22% (12)	36.49% (27)	3.74	74
SkiROS2	1.35% (1)	9.46% (7)	22.97% (17)	28.38% (21)	6.76% (5)	31.08% (23)	3.43	74
AS2FM by CONVINCE	4.11% (3)	15.07% (11)	20.55% (15)	13.70% (10)	5.48% (4)	41.10% (30)	3.02	73

Average rating: 3.72

Q9

How easy was it to implement error handling and recovery behaviors in each tool?

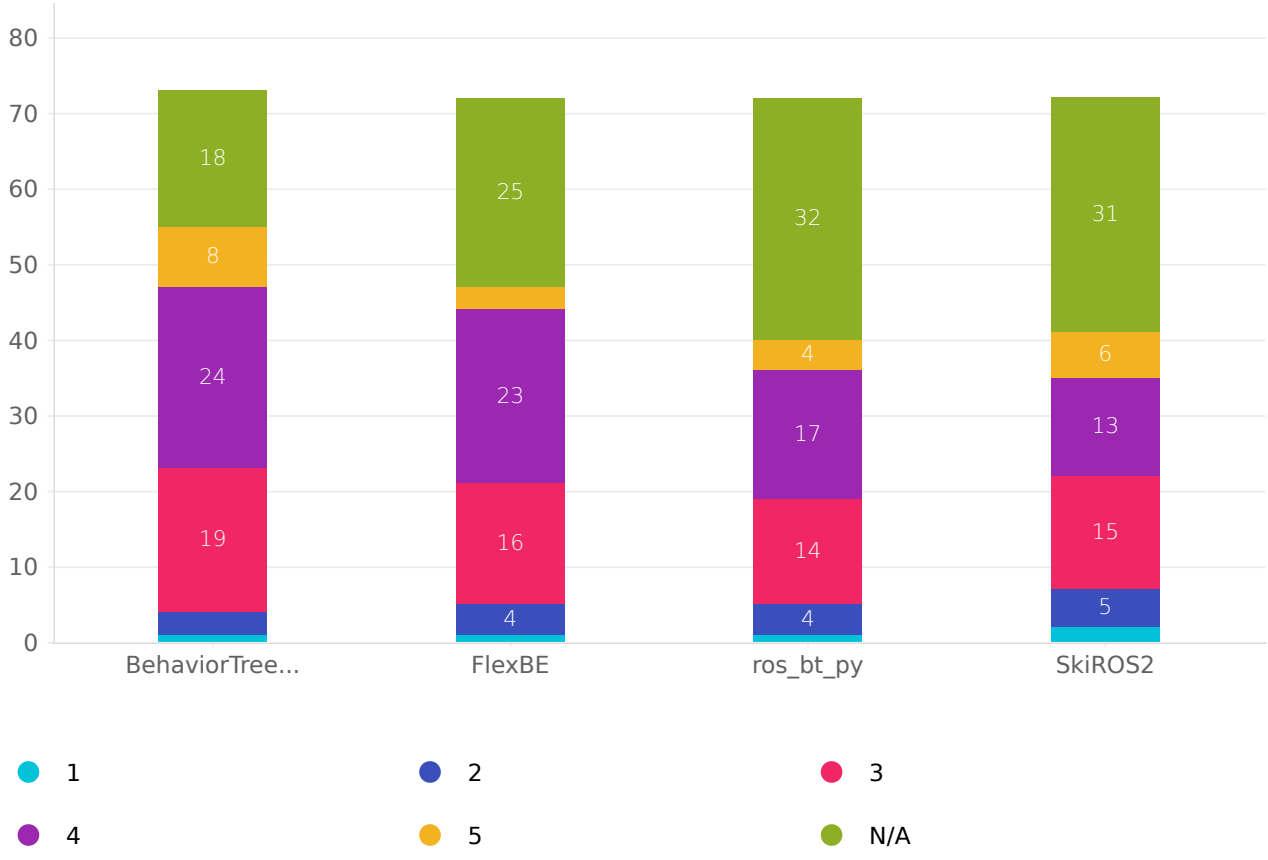


Row	1 (Very difficult)	2	3	4	5 (Very easy)	N/A	Average rating	Response count
BehaviorTree.CPP	1.33% (1)	4.00% (3)	21.33% (16)	33.33% (25)	17.33% (13)	22.67% (17)	3.79	75
FlexBE	1.35% (1)	1.35% (1)	17.57% (13)	31.08% (23)	10.81% (8)	37.84% (28)	3.78	74
ros_bt_py	1.35% (1)	5.41% (4)	22.97% (17)	13.51% (10)	9.46% (7)	47.30% (35)	3.46	74
SkiROS2	10.81% (8)	8.11% (6)	17.57% (13)	10.81% (8)	2.70% (2)	50.00% (37)	2.73	74

Average rating: 3.50

Q10

How effective is the tool in adapting to changes in the environment or task requirements?

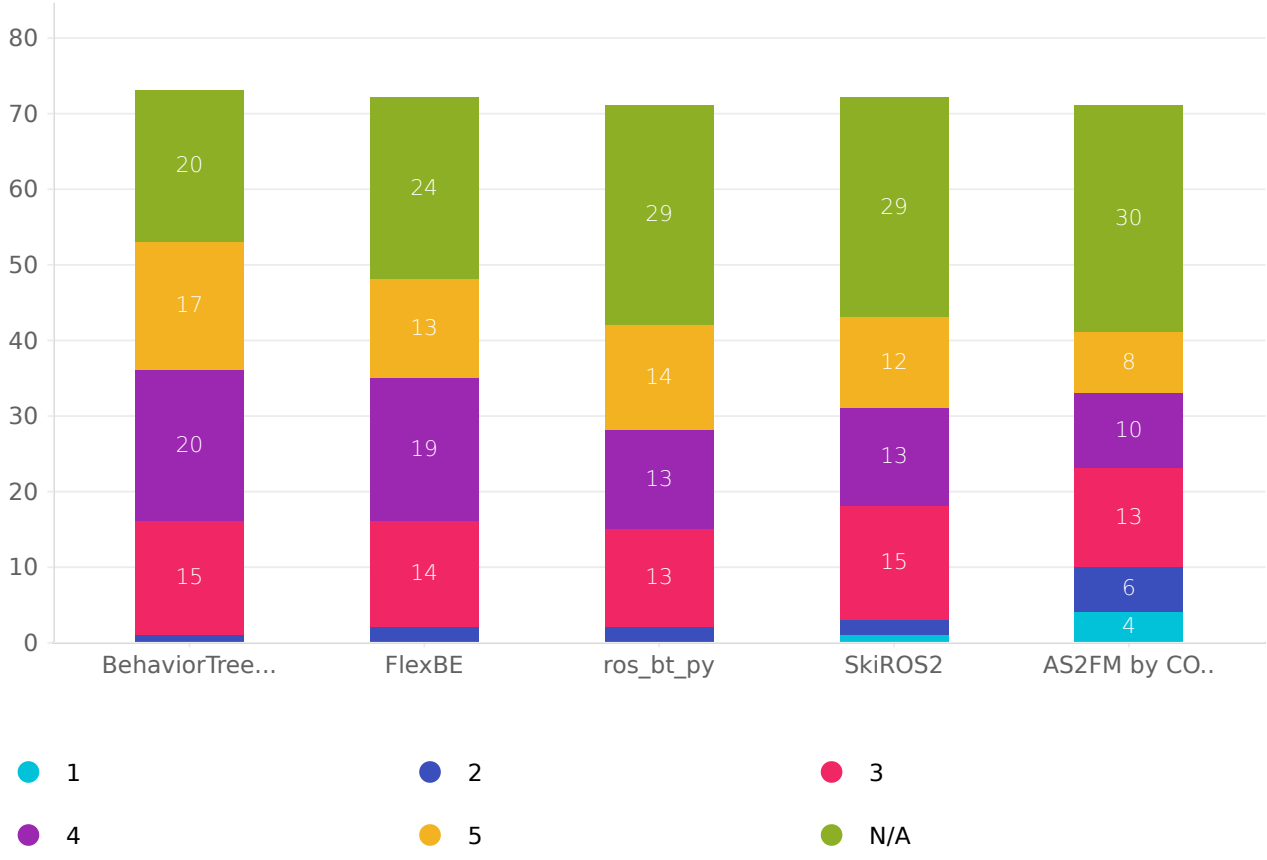


Row	1 (Not effective at all)	2	3	4	5 (Very effective)	N/A	Average rating	Response count
BehaviorTree.CPP	1.37% (1)	4.11% (3)	26.03% (19)	32.88% (24)	10.96% (8)	24.66% (18)	3.64	73
FlexBE	1.39% (1)	5.56% (4)	22.22% (16)	31.94% (23)	4.17% (3)	34.72% (25)	3.49	72
ros_bt_py	1.39% (1)	5.56% (4)	19.44% (14)	23.61% (17)	5.56% (4)	44.44% (32)	3.47	72
SkiROS2	2.78% (2)	6.94% (5)	20.83% (15)	18.06% (13)	8.33% (6)	43.06% (31)	3.39	72

Average rating: 3.51

Q11

How well did each tool handle integration with other ROS2 components and systems?

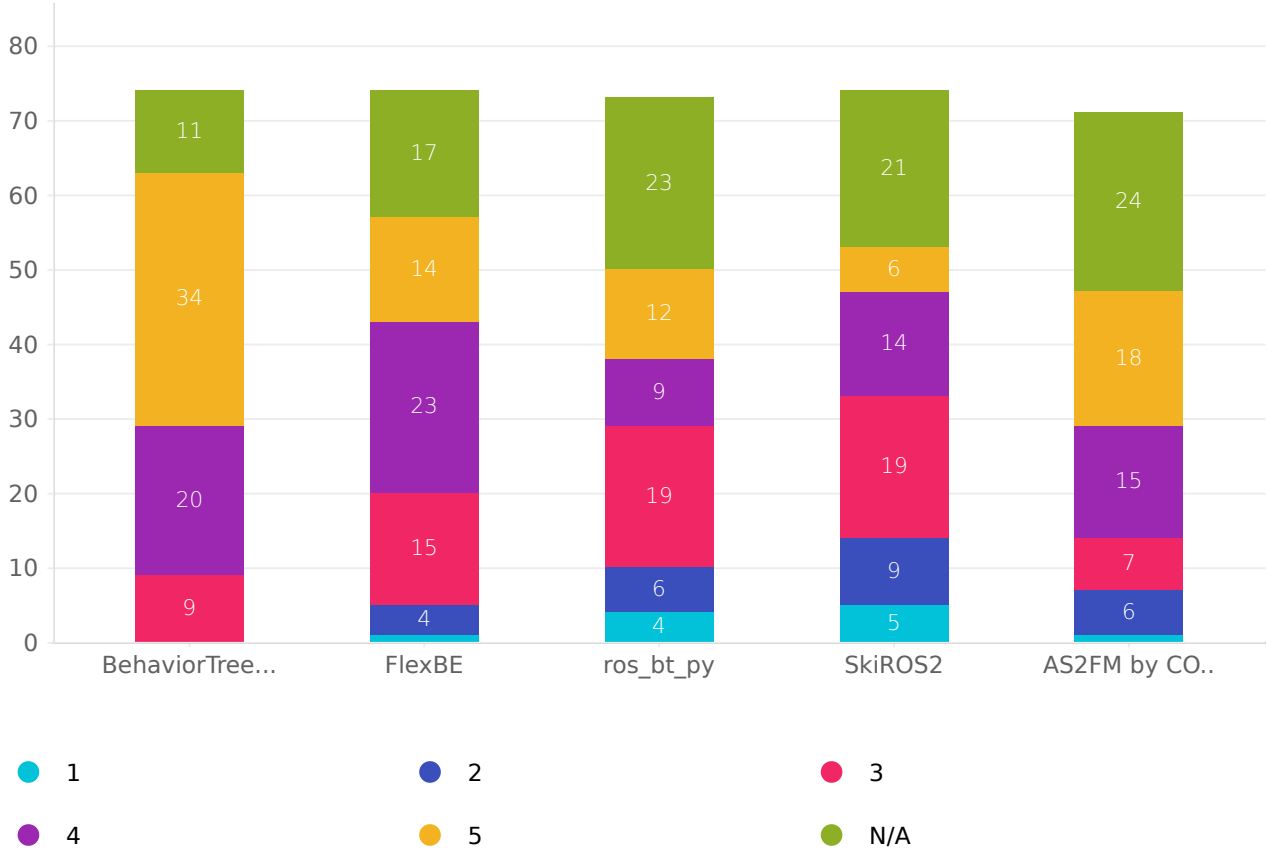


Row	1 (Not well at all)	2	3	4	5 (Very well)	N/A	Average rating	Response count
BehaviorTree.CPP	0.00% (0)	1.37% (1)	20.55% (15)	27.40% (20)	23.29% (17)	27.40% (20)	4.00	73
FlexBE	0.00% (0)	2.78% (2)	19.44% (14)	26.39% (19)	18.06% (13)	33.33% (24)	3.90	72
ros_bt_py	0.00% (0)	2.82% (2)	18.31% (13)	18.31% (13)	19.72% (14)	40.85% (29)	3.93	71
SkiROS2	1.39% (1)	2.78% (2)	20.83% (15)	18.06% (13)	16.67% (12)	40.28% (29)	3.77	72
AS2FM by CONVINCE	5.63% (4)	8.45% (6)	18.31% (13)	14.08% (10)	11.27% (8)	42.25% (30)	3.29	71

Average rating: 3.79

Q12

How satisfied were you with the speed and responsiveness of each tool during execution?

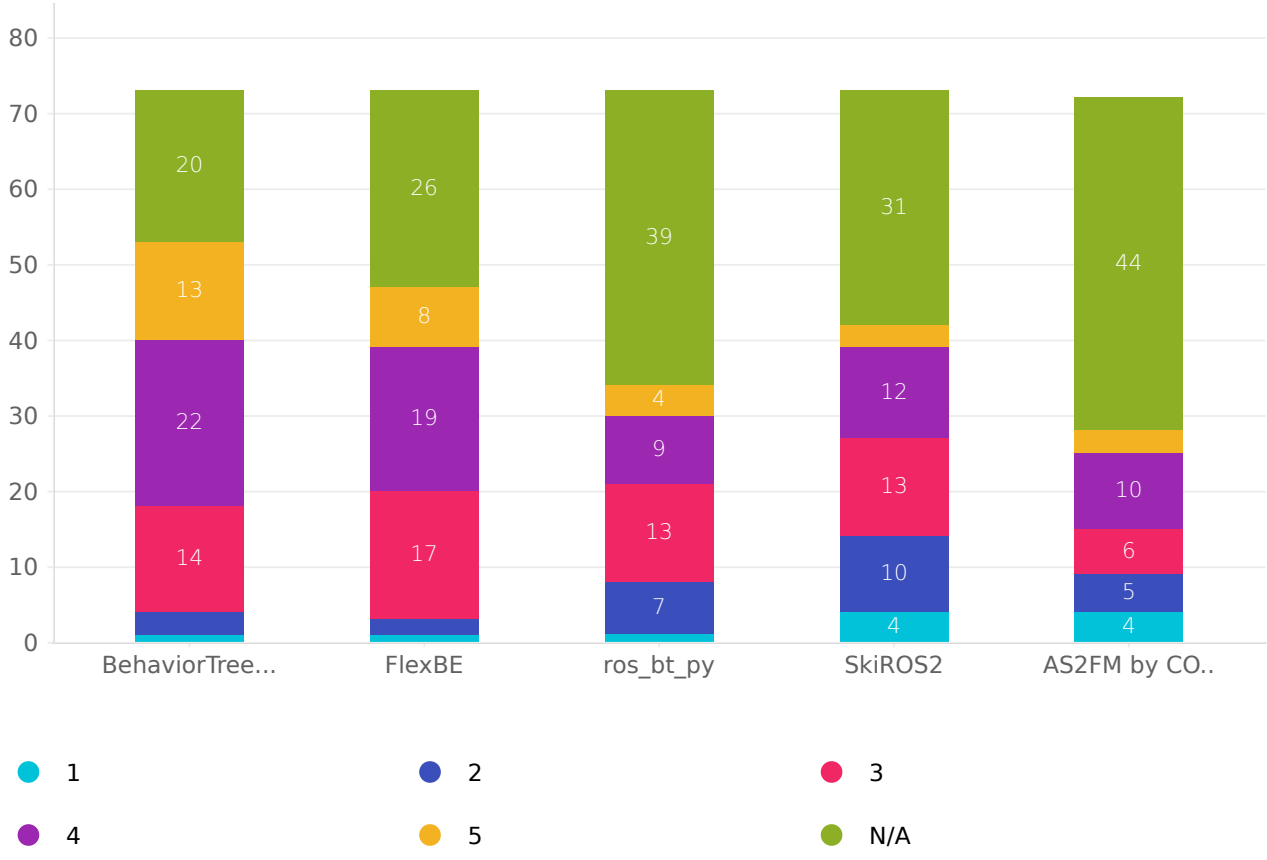


Row	1 (Not at all satisfied)	2	3	4	5 (Very satisfied)	N/A	Average rating	Response count
BehaviorTree.CPP	0.00% (0)	0.00% (0)	12.16% (9)	27.03% (20)	45.95% (34)	14.86% (11)	4.40	74
FlexBE	1.35% (1)	5.41% (4)	20.27% (15)	31.08% (23)	18.92% (14)	22.97% (17)	3.79	74
ros_bt_py	5.48% (4)	8.22% (6)	26.03% (19)	12.33% (9)	16.44% (12)	31.51% (23)	3.38	73
SkiROS2	6.76% (5)	12.16% (9)	25.68% (19)	18.92% (14)	8.11% (6)	28.38% (21)	3.13	74
AS2FM by CONVINCE	1.41% (1)	8.45% (6)	9.86% (7)	21.13% (15)	25.35% (18)	33.80% (24)	3.91	71

Average rating: 3.75

Q13

How effective was the tool in the introspection/debugging of programmed behaviors?



Row	1 (Not effective at all)	2	3	4	5 (Very effective)	N/A	Average rating	Response count
BehaviorTree.CPP	1.37% (1)	4.11% (3)	19.18% (14)	30.14% (22)	17.81% (13)	27.40% (20)	3.81	73
FlexBE	1.37% (1)	2.74% (2)	23.29% (17)	26.03% (19)	10.96% (8)	35.62% (26)	3.66	73
ros_bt_py	1.37% (1)	9.59% (7)	17.81% (13)	12.33% (9)	5.48% (4)	53.42% (39)	3.24	73
SkiROS2	5.48% (4)	13.70% (10)	17.81% (13)	16.44% (12)	4.11% (3)	42.47% (31)	3.00	73
AS2FM by CONVINCE	5.56% (4)	6.94% (5)	8.33% (6)	13.89% (10)	4.17% (3)	61.11% (44)	3.11	72

Average rating: 3.42

Section 5: Strengths and limitations

The following questions are open-ended, and we would appreciate your feedback on the strengths and drawbacks of each tool to help us improve them in the future.

Q14

AS2FM by CONVINCE

What do you consider to be the key **strengths** and **limitations** of this tool? Please provide details on what aspects you find most useful or beneficial, as well as any suggestions for improving the tool.

1. Seems like we are testing a model the was created with this new language, and not the actual system that will run.
2. Strength: make sure things work as intended Limitations: Have to do another version of your code in another language (or atleast that is how it seems).
3. Seems very powerful for verifying that a particular system is modelled properly for what you are trying to achieve. One drawback, unless I am missing something, it requires you to model your behaviors accurately in scxml and jani, which seems prone to error?
4. strength: get information on reliability which would otherwise not be available limitation: hard to implement the syntax
5. Strength: Being able to validate assumptions and see if things check out or if the program needs to be adjusted. Limitation: Seems to be quite some work to convert, and hard to judge if everything is convertable from the small toy examples we saw.
6. Strengths: Simulation based robotic problems Limitations: Readability and complexity of the tool
7. Not familiar enough to reply such a complicated question. Workshop just gave me a glimpse of deliberating techniques.
8. Strength - No other tool like this out there. Limitations - having to write nodes in scxml
9. It seemed simple to use.
10. I don’t really understand / see an usage for the system
11. Really useful tool to check models. Could be bit more advanced workshop though
12. Strengths: Check coverage and better test your application Limitations: If coverage is not as expected, there is poor traceability of the defect (is it in the scxml file or in the actual implementation of the skill?)
13. It seems very strong in testing models faster full coverage tools. But i did not quite grasp how useful this is for creating behaviours
14. Pros: Very fast and efficient usable numbers, defining the quality of models. As stated in presentation, these are quite hard and slow to calculate manually - and invaluable for evaluating solutions. Cons: For now, a lot of setup is required before meaningfull tests can be performed. This means that the tool is best used to evaluate a model when it is near completion. These kinds of insights could be more valuable earlier in the development process.
15. Limitation: a) Not very mature yet b) While it will help you know that a failure case exists, it's not clear to me if you will be able to see what sequence of events will lead to the failure case. This would mean it would be really hard to solve the problem, even though you know it exists. Strengths: It will cover far more scenarios /edge-cases than one could realistically write otherwise.
16. I think the strengths are the way the xml files can be configured easily to define a set of tasks. This requires you to become familiar with a different way of implementing logical operators which makes this tool not intuitive unless you’re familiar with that system. Once you’re familiar then modifying xml files is easy

Q15

SkiROS2

What do you consider to be the key **strengths** and **limitations** of this tool? Please provide details on what aspects you find most useful or beneficial, as well as any suggestions for improving the tool.

1. Too complex at start.
2. Strengths: Efficient autonomy in a well defined modelled world, where you can add skills as you go to expand the possibilities. Limitations: Requires you to be able to model everything well, and is a bit over overhead for problems that are not complicated enough.
3. It seems very intuitive to describe behaviors at a very high level
4. strength: find a plan automatically based on skills limitation: hard to implement the syntax
5. Strength: Very cool with auto-generation that made it easy to configure and change show. Limitation: I feel a bit unsure on what is generated underneath at times.
6. I dont understand what this tool solve really. User interface
7. Not familiar enough to reply such a complicated question. Workshop just gave me a glimpse of deliberating techniques.
8. Strengths - integration with PDDL Limitations - hard to debug pddl
9. The presentation was very helpful in understanding how to use this tool. Otherwise, I think I would have been lost. It seems very slow when executing the plan, which makes me wonder how it would work well when implemented on real hardware.
10. Easy to use maybe limited respect the other tools presented today
11. Very useful tool. Seamless to use.
12. Strength: Automatically determine task schedule Limitation: For simple applications is it too much overhead?
13. Behavioral tree planning
14. Very difficult to debug when planner fails Should save the plan for debugging execution Planner is very slow
15. seems very powerful for autonumous systems, but I assume it becomes rather slow for problems that are reallife sized and contain more then 5 rooms and 4 objects.
16. Pros: Great way of using knowledge based logic to dynamically creating plans. This keeps solutions from having to have backup plans pre-defined. Cons: Setting up environment can be an elaborate tasks, taking valuable time. It is therefore not always useful for simple tasks.
17. I was late to the workshop and did not get this exposure.
18. I really enjoyed that the world, skills are defined using a python file to create links between components in the world. I think the hardest part is understanding how the skills get transformed into decisions once the planning system is running. I think seeing a GUI with the process running would help understand what is happening. However when you’re familiar with the world and the skills a robot makes then making tasks is easy.

Q16

BehaviorTree.CPP

What do you consider to be the key **strengths** and **limitations** of this tool? Please provide details on what aspects you find most useful or beneficial, as well as any suggestions for improving the tool.

1. Error handling had been a struggle when dealing with complex trees .
2. Strengths: Modularity and debugging and understanding what is going on and how things are connected (Groot2 looks great). Limitations: Not much, for me it's only the fact that It is in C++. And of course the general things that it requires more overhead compared to simpler finite state machines or other technologies.
3. Safety critical Realtime
4. strength: easy to maintain and extend or modify limitation: unintuitive at first because , Free GUI only supports 20 Nodes
5. Grouping problems into subtrees was very nice, and and I liked that it was an open choice og GUI. I found it very hard to do more complex tasks than simply scheduling tasks after each other. Especially task 4 was hard to implement with this tool. Could not figure out how to switch to "charginge mode" when battery was empty, and then continuing instead of restarting on the mission afterwards.
6. Strength: Seems very cool to setup and configure things via the GRoot2. Limitation: Can be expensive to get going with Groot2 to debug with it. In ROS1 I use py_trees at work which makes Behaviour trees far preferred for me over FSMs. So I assume the things for ros_bt_py and BehaviourTree.CPP holds true for both and I would like to use both as I like behaviour trees. Like BehaviourTree, I like to have a GUI to configure the nodes and structure with, instead of writing a lot of code to create the structure of my tree which I have to in ROS1 for py_trees.
7. Strengths of this tool is easy to integrate any system. Limitations of this tool is replay properties.
8. Strengths: Modularity and a wide variety of nodes to choose from.
9. I think this is very important and it made sense during the presentation, but I was very lost on how to implement this.
10. Groot2 should became inside the package like the other tool
11. Very good tool. Liked the groot GUI!
12. Some way outside of blackboard design to share data would be nice
13. Strengths: Good visualization and debugging of the task execution
14. Easy to use gui interface, very user friendly
15. The GUI was good and intuitive. It would be very nice if it is possible to autocomplete things such as navigation targets and objects (to use examples from the exercises) when filling out parameters
16. Seems pretty mature. The GUI and XML files felt by far the nicest out of all the tools today. Very nice!
17. Strength: The tooling, especially drag and drop editor seems extremely useful, especially compared to how we rendered our trees in py_trees. I do wonder how it will handle some of the very large trees that we had Limitation: We use py_trees before, I found that using code to both create primitives, but also hook them up to each other was beautifully flexible, it did not predetermine how things should be done, especially once we started deviating from usual use cases(we may have abused the API a bit). I am worried that BehaviorTree.CPP would feel rigid in comparison.
18. Creating scripts using Behaviour trees is made simple with the UI which makes this tool very powerful.

Q17

FlexBE

What do you consider to be the key **strengths** and **limitations** of this tool? Please provide details on what aspects you find most useful or beneficial, as well as any suggestions for improving the tool.

1. Nice in general for prototype and quick iterations. Really nice tool.
2. Strengths: The fact that the operator is integrated so well, makes it cool for those specific use cases and to just develop and debug as a developer. Limitations: The same thing though, if i need to use it, I feel like i kinda have to have a user, unless i just use full autonomy. But then again, full autonomy is based on "suggestions" and such. So in my production environment with no operator, it might seem weird to use. But then again, it might just be my limited use and understanding of the tool.
3. Not safety critical
4. strength: good gui, setting of autonomy levels limitation: can become hard to maintain compared to behavior trees
5. Strength: For concurrency it seems nice that something can take priority (charge battery) without having to design a complex behaviour tree to take into account that you need to charge at some point. Limitation: Seems very hard to read and configure. Where a behaviour tree is quite easy to keep track of what happens and what will happen in which order.
6. Limitations of this tool is python and user interface is not easy to understand
7. It can provide high level interface control for robotic systems.
8. The user interface feels a bit more dated and harder to get into than the other tools
9. Strengths - good UI, rich capability Limitations - accidentally clicking on transitions before execution is complete put system in strange state
10. This included a nice GUI that was easy to use and a good guide to follow. It was also great to have a "collaborative autonomy" implemented.
11. I think that the tool has an enormous power, maybe is a little messy at the first impact
12. Good tool addressing problems of BTs. Gui could be better.
13. Fsms are always easiest to reason about.
14. Well understood outline for flow of the robots states
15. You need to save all the time Changes on state should be persistent without pressing Apply
16. Easy to debug.
17. I don't know, I joined late
18. Limitations: not good with high levels of complexity / connection Strengths: good at collaborative autonomy
19. I really enjoyed the graphical UI for defining the behaviours of the world and the combination of state machines and behaviour trees for creating tasks. I would be curious if you could include more specifics on what’s required when trying to save behaviours because this made the design process a bit confusing.

Q18

ros_bt_py

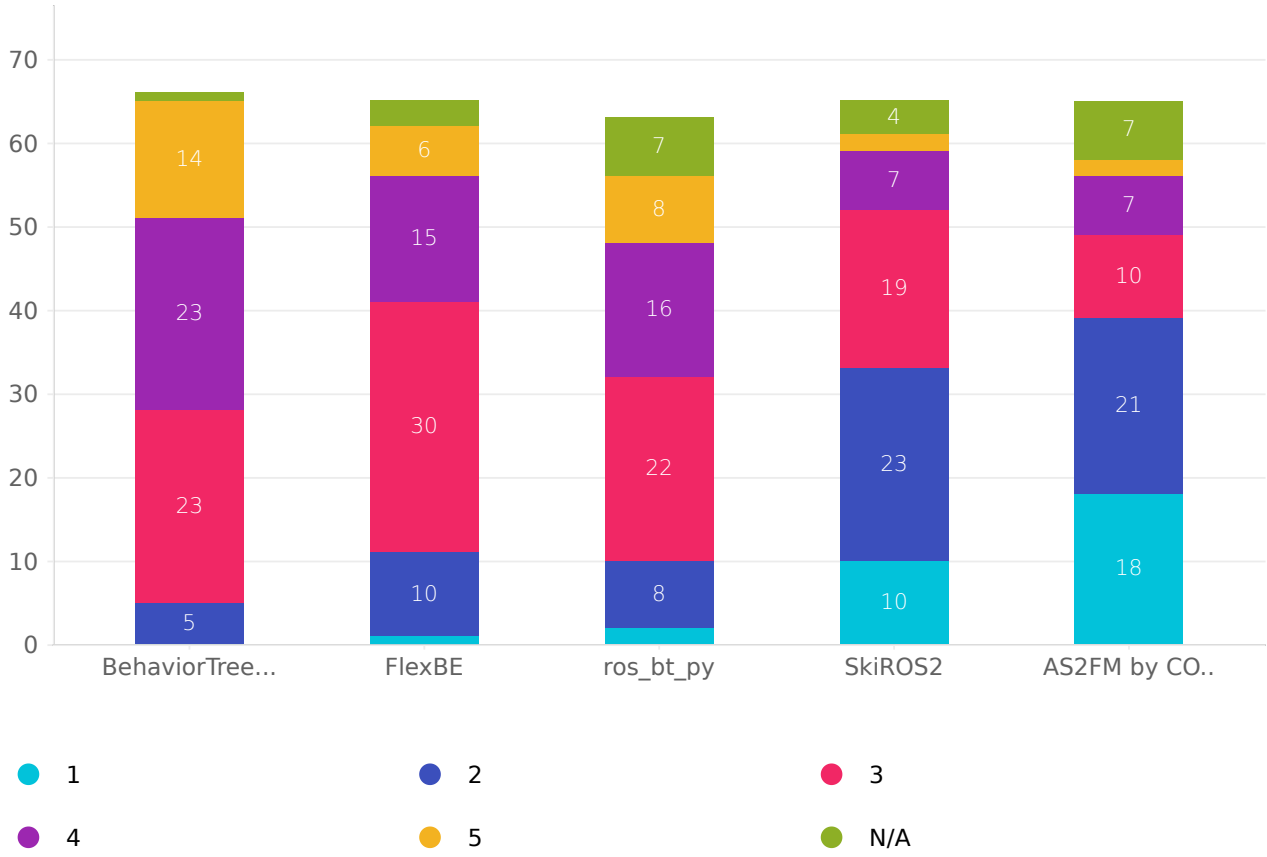
What do you consider to be the key **strengths** and **limitations** of this tool? Please provide details on what aspects you find most useful or beneficial, as well as any suggestions for improving the tool.

1. Looks good in general, UI was a bit buggy. Node should not crash due to errors in user interface.
2. Strengths: Great way to quickly get something to run in a ros2 environment. Limitations: Requires a very well ros implemented system, and it might require a bit of overhead when only using ros2 based functionality.
3. strength: nice webgui limitation:
4. I didn't get to use this, though I would say that it seems really good. In ROS1 I use py_trees at work which makes Behaviour trees far preferred for me over FSMs. So I assume the things for ros_bt_py and BehaviourTree.CPP holds true for both and I would like to use both as I like behaviour trees. Like BehaviourTree, I like to have a GUI to configure the nodes and structure with, instead of writing a lot of code to create the structure of my tree which I have to in ROS1 for py_trees.
5. Limitations of this tool is python
6. N/A
7. Easy web interface, maybe just to fix some usability issue: add suggestion / text to the icon and a better search motor
8. Tool is easy to use. Workshop is bit all over the place
9. The data sharing feature seems really nice
10. Simple interface to create tasks plans that is fully integrated with ros
11. Hard to use interface
12. It seems to be easy to use which is also due to it's python base, but it's probably too slow for many real world robotics tasks (?)
13. I joined late so I don't know
14. I really liked the UI however I didn’t have enough time to try and run the examples because the trace back threw errors which I had no time to double check

Overall experience

Q19

How easy do you think it would be for a novice to learn the following tools?

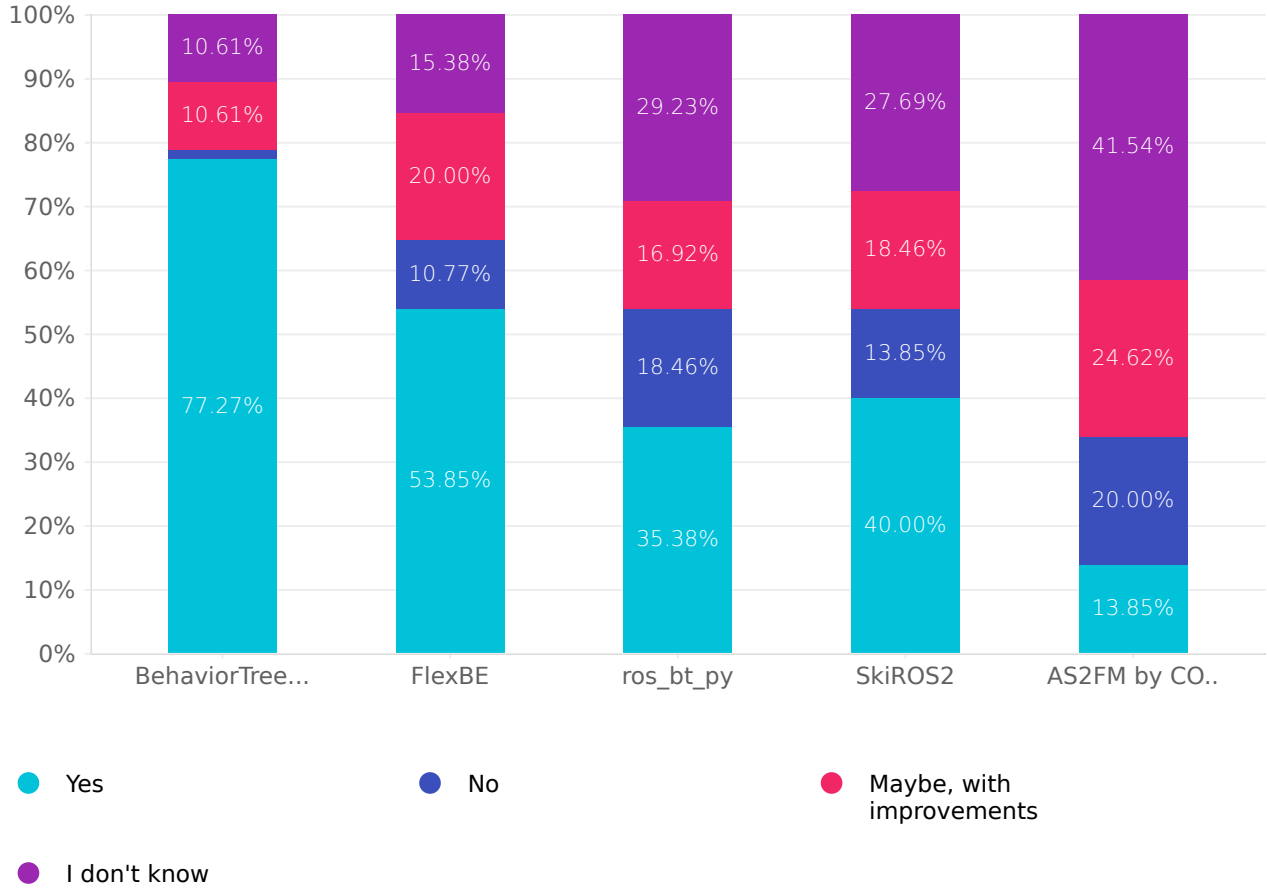


Row	1 (Very difficult)	2	3	4	5 (Very easy)	N/A	Average rating	Response count
BehaviorTree.CPP	0.00% (0)	7.58% (5)	34.85% (23)	34.85% (23)	21.21% (14)	1.52% (1)	3.71	66
FlexBE	1.54% (1)	15.38% (10)	46.15% (30)	23.08% (15)	9.23% (6)	4.62% (3)	3.24	65
ros_bt_py	3.17% (2)	12.70% (8)	34.92% (22)	25.40% (16)	12.70% (8)	11.11% (7)	3.36	63
SkiROS2	15.38% (10)	35.38% (23)	29.23% (19)	10.77% (7)	3.08% (2)	6.15% (4)	2.48	65
AS2FM by CONVINCE	27.69% (18)	32.31% (21)	15.38% (10)	10.77% (7)	3.08% (2)	10.77% (7)	2.21	65

Average rating: 3.01

Q20

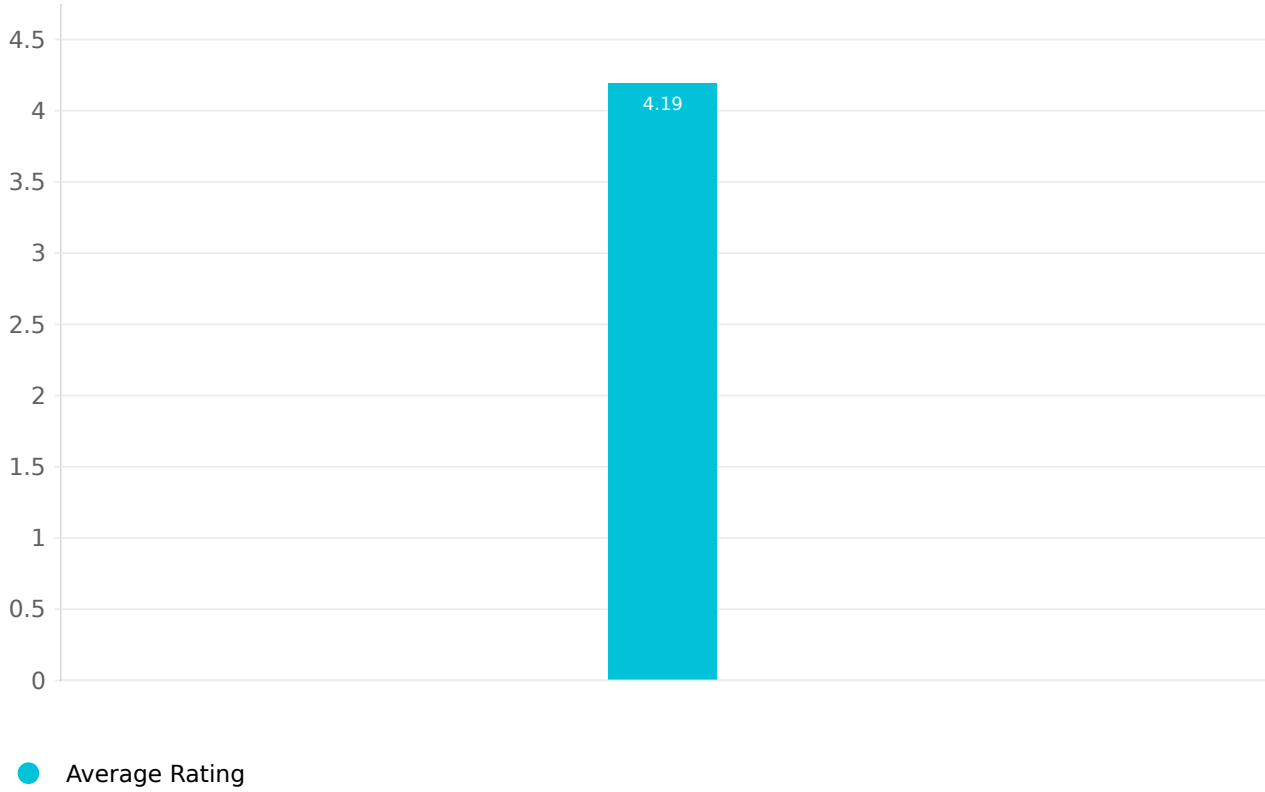
Would you consider using or recommending the following tools in your future projects?



Row	Yes	No	Maybe, with improvements	I don't know	Response count
BehaviorTree.CPP	77.27% (51)	1.52% (1)	10.61% (7)	10.61% (7)	66
FlexBE	53.85% (35)	10.77% (7)	20.00% (13)	15.38% (10)	65
ros_bt_py	35.38% (23)	18.46% (12)	16.92% (11)	29.23% (19)	65
SkiROS2	40.00% (26)	13.85% (9)	18.46% (12)	27.69% (18)	65
AS2FM by CONVINCE	13.85% (9)	20.00% (13)	24.62% (16)	41.54% (27)	65

Q21

Overall, how would you rate the workshop on a scale of 1 to 5 stars, with 5 being the most useful and 1 being the least useful.



Number of stars	Response percent	Response count
★	0.00%	0
★★	4.76%	3
★★★	11.11%	7
★★★★	44.44%	28
★★★★★	39.68%	25
Please share which area you find most or least helpful/enjoyable.		19

Average rating: 4.19

Please share which area you find most or least helpful/enjoyable.

1. A great overview and with great docker setup and problems/simulations. Very nice to just get going.

2. Mix of the different approaches in the theory part is somehow confusing for novices. 1 Tool -> Experiments, next tool -> experiments would be easier to capture and helps avoiding mixing up concepts.
3. I only got tile to try out one of the tools. Would have liked more time for implementation and less talks. Maybe the task also should have been smaller or more compressed so that we quicker can try out different tools. I generally liked the simulator, but some things made it hard to generalize, such as you needing to know the name of the hallway to open it.
4. I found it really nice to see several different tools that showed how a problem can be solved. And how there were several problems that extended on each other and we could see and play with a simple and light-weight simulator.
5. I learned a lot more deliberation tools than i did before this.
6. AS2FM missed some example for actual use in a ros project at the end, maybe including unit tests
7. Tutorials were really simple and good. None of the workshops were too hard to follow. Exposure to new libraries were great.
8. Workshop was a good overview over available deliberation frameworks, also excellent preparation of the tooling and the Docker container (that was no small amount of work).
9. This was very fun and interesting! The hand-on interaction was great!
10. I would like a little more initial info on how to run things.
11. Very good overview of technologies! Only remark there was not much time to experiments with the code and solve the tasks
12. The problems were super enjoyable and how each technology applied to it
13. It was a bit overwhelming with this many tools to solve the same problem and having to explore all of them didnt allow alot of time to understand any of them on more than a surface level. I would have liked the workshop to focus on a single tool and then go more in depth. Good presentations from everyone and very nice teaching material and problems
14. I don't know if it was the workshop or my attention span, but it started really strong and then became harder and harder to follow towards the end. But I learned a lot and enjoyed it very much. Thank you
15. To much things to try. Each tool would require a full day workshop.
16. Was a great overview on the various approaches and implementations on deliberation.
17. 3
18. Many more slides could probably have less words. At times, it can be hard to understand as a beginner with so much information and little frame of reference. Then again, experience and expected audience differs.
19. I really enjoyed seeing how each technology would approach this problem. I would be curious what would happen if you combined technologies to solve the problems given. I think this would help highlight the strengths/weaknesses of each tool

Q22

Any additional comments or insights you would like to share with the workshop organizers?

1. I'm slightly overloaded with information, and I felt I really had to choose what I wanted to try out. But then i have the setup for later if i want to try out more, so I guess it is pretty good still.
2. Maybe better to dive hands on deep into one framework
3. Thank you
4. too little time to work on the problems after struggling with setting up the tools in the beginning
5. Some tools seemed really easy, but I also feel that was maybe because a lot was abstracted away (looking at SkiROS and CONVINCE), while FlexBE might be good some of the examples handled way too much and thus looked to complex for me to want to start working on. BehaviourTree.CPP seemed to trade-off this quite well and showed something simple that we could then expand on. However, I don't know how much code Davide Faconti had made behind the scenes for some of the behaviours, But, these complaints would maybe make this way more than a 1 day workshop and instead be a multi day/week workshop to dig in properly to each of these things. Furthermore, I have used behaviour trees in my day-job so maybe I am a bit biased to working with that.
6. I feel like the time until the first hands on part was too long Using only discord for sharing slides is a bit annoying, I would prefer links/PDFs on GitHub or the roscon homepage
7. Way too much with so many frameworks in one workshop. Explanations and especially commands to be run went by too fast, could not keep up. I was able to fiddle a bit with one of the tools, but really did not have time to get anywhere with the rest.
8. Many software stacks have the "air" of being developed by universities during grant-funded projects, and being mostly abandoned at the end of financing. Also the usability and focus often seems of seems to be on generating papers and in generally serve academic purposes making them less than usable in industrial settings.
9. Maybe doing all the theoretical part in the morning and the coding part in the afternoon…for the coding for present the problem leave like 30min and present a solution
10. Maybe split the first work alone session into two half hour session. It was a lot of information for me as a newbie and would have been nice to try out the behavior trees before going to the HFSM
11. The workshop was great! The only issue is that even in simple examples as the problems proposed, there was not enough time to implement the solutions.
12. We need more tinkering time I would remove the last talk and use the time for working in the problems
13. Thanks a lot! Great workshop.
14. I found it sometimes hard to do the activities and still follow the talk at the same time
15. In the discussions, behavior trees vs. state machines came up. In practice I found behavior trees far more practical to work with than FSMs. Main benefits were the re-use of subtrees when we building new robots / behaviors and how one can just gradually layer on new features and complexity often without messing with subtrees we built already.
16. Would have liked it to be more fun/fewer monotonous commands slide after slide.
17. It would be great if the speakers walked through the first problem set till everyone has it working before speeding up.
18. Great workshop! Thanks!
19. I enjoyed all the presentations, I think there wasn’t enough time to fully dig deeper into the technologies. It would be nice if we could have some of the slides prior to the session for us to play around with some of these tools beforehand.