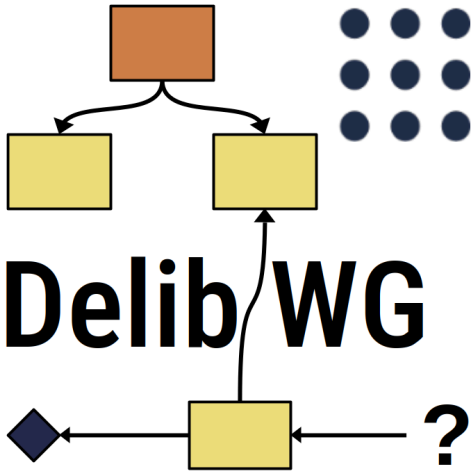
The ROS Deliberation Community Group



This document is intended as the main landing page for the community group. It also contains agendas and minutes of our meetings.

Leads

Christian Henkel
Contact: Christian.Henkel2@de.bosch.com
Sebastian Castro
Contact: sebas.a.castro@gmail.com

Mission

We started this community group to discuss which tools and frameworks exist in the ROS community to design a robotics deliberation layer. Deliberation is the layer of a robotics software architecture that orchestrates the robot's skills and features in order to reach a goal autonomously. This is sometimes also called the mission control or planning layer.

In the group, we want to exchange knowledge about software tools in the deliberation layer that have some connection to ROS. This learning starts by identifying which of these tools and frameworks exist right now and how they can be utilized successfully in robotics systems. But we also want to identify synergies in order to improve their interchangeability. This also includes the identification of common guidelines for the design and application of these systems.

Products

As a working group, we have produced and are producing the following deliverables:

- To get an overview on what is out there, a list of existing tools, SW resources for deliberation in ROS 2: https://github.com/ros-wg-delib/awesome-ros-deliberation
- A workshop on ROS 2 Deliberation Technologies, presented at ROSCon 2024. This has since been converted to an evolving examples repository: https://github.com/ros-wg-delib/examples
- A workshop on reinforcement learning for deliberation, presented at ROSCon 2025: https://github.com/ros-wg-delib/rl_deliberation

Meeting Invitation

If you want to receive an invitation to the working group meetings, please join the ros-wg-deliberation Google group.

Working mode and access rights

Please add points that you want to discuss, or have discussed during the meeting, to the agenda found later on in this document.

All members of ros-wg-deliberation have rights to suggest changes. If you are attending the meeting regularly and would like to help moderate the changes, we can give you edit rights. But for security reasons, we don't want to give them to all group members by default.

Discourse

You can find publications regarding this working group on Discourse under the Deliberation Community Group.
NOTE: Older posts were found under the tag wg-delib.

Discord

Please feel free to join our discord server: https://discord.gg/Qt9veJ6WJx

26-07-06

Deliberation at ROSCon 2026 (Presentation acceptance on June 9)

26-06-01

Tool Spotlight (5 min)
Ports and XML Parsing in PyTrees
by Sebastian Castro and Jennifer Buehler

Presentation (30 min + 10 min discussion)
Runtime Verification: Offline and Online Monitoring Tools

- **DLR-FT/TBT-Segmentation: This tool allows to segment a trace of events using a temporal behavior tree specification**
- **DLR-FT/RTLola-ROS2-Adapter: Automatically exposes ROS 2 topics to CISPA's RTLola interpreter without manual overhead.**

by Sebastian Schirmer from German Aerospace Center (DLR)

26-05-04

Tool Spotlight (20 min)
B2Xklaim: https://github.com/khalidbourr/B2XKlaim
by Khalid Bourr (University of Trieste)

- Abstraction of multi-robot system
- With DSLs
- X-Klaim programming language (https://link.springer.com/chapter/10.1007/978-3-030-61470-6_22)
- High-level notation with Business Process Model and Notation (BPMN)
- Two bridges:
    - ROS2 **-** X-Klaim: https://link.springer.com/article/10.1007/s10009-023-00727-w
    - X-Klaim **-** BPMN: https://link.springer.com/article/10.1007/s10009-025-00832-y
- Use case: Drone rover coordination for scouting mission

Presentation (30 min + 10 min discussion)
AutoAPMS: The scalable behavior development framework for ROS 2:
https://github.com/AutoAPMS/auto-apms
https://github.com/AutoAPMS/auto_apms_studio
by Robin Müller and Ewald Hild from TU Darmstadt

- Use Case: Behavior definition for drones, e.g. tracking and avoidance of obstacles

- Motivation: Modularity, maintainability
- Other solutions found to lack versatility, dev-x and scalability
-

26-04-13

Discussion (slides again)

- **Which programming languages do you prefer to use for deliberation?**
    - Francisco: Programming language is not important since the time scale tends to not be super critical for decision-making. Use the language that helps you.
        - “I like programming in C++, Python has betrayed me many times”
    - Michele: With interpreted languages, you get all the errors at runtime (e.g. with corner cases). Would prefer to have compile-time errors.
        - Ease of programming is less of an issue now that LLMs are useful.
    - Christian: Quicker turnaround time for prototyping if using Python?
    - Maroussa: As a newcomer, gravitating towards Python because of the lower barrier to entry.
    - Martin: Also dependent on your product’s lifecycle, probably more tolerant to interpreted languages earlier in the lifecycle for speed of prototyping, but may need to transition later due to e.g. hard real-time constraints.
        - AI assistance can be very good for porting code to other languages
        - Higher in the layer, it’s not because Python is slow, it’s because it can be unsafe
    - Khalid: Also use Java (solid concurrency primitives, portability due to JVM) for multi-robot coordinate applications.
        - Of course, Java is not supported by ROS, have to depend on external libraries. Was using rosbridge, but now using jros2 (https://github.com/ihmcrobotics/jros2).
        - Apparently there is a ros2 Java! https://github.com/ros2-java/ros2_java
    - Daniel: Mainly Python, easy to implement more complex logic. Other key (performance-critical?) components can still be in the C++ side. Mostly interacting with ROS servers (services/actions)
- **What are the benefits of declarative and imperative languages (respectively) for deliberation?**
    - Christian: Easy to prototype with declarative, if the underlying skills are implemented properly
    - Michele: Prefer the declarative, but anyways you need to actually implement the underlying behaviors.
        - With the “behavior designer hat”, go full declarative because you don’t care about the implementation. You’re reasoning at a higher level.
        - With the “skill implementation hat”, implementations should probably be more imperative?
    - Francisco: There are many approaches for auto-generating the declarative part (e.g. an LLM that can spit out structured XML).

    - Martin: If both approaches worked equally well, most engineers would choose declarative because it reduces cognitive burden. Harder to debug outside of toy examples.
    - Christian: The core principle is hard abstraction. You are forced to separate your concerns. If you have nicely implemented skills, you can transfer to multiple abstractions. However, it's more effort.
    - Martin: Composition of skills is simpler in declarative
    - Sebastian: Compare DSL, you must be more careful with abstraction, e.g. in pytrees you could mix concerns a lot more
    - Khalid: Declarative has appeal when moving to visual programming. Useful to compile declarative code to imperative so you can put your breakpoints in there.
    - Maroussa: What about an "ethical" layer, example expressing high-level preferences for behavior? Or validation?
        - Sebastian: for example retroactive parsing of behaviors is easier on a declarative language
        - Martin: Reminds me of the claude code regex search on user prompts https://www.reddit.com/r/HowToAIAgent/comments/1s8ytdr/claude_codes_source_just_leaked_and_one_thing/
        - Christian: Must probably be done on a symbolic layer, therefore a declarative deliberation layer seems to be more useful
        - Francisco: This is even higher level than the deliberation we're talking about, e.g, validating plans, enforcing symbolic constraints, etc.
        - Khalid: Could even mix multiple layers, e.g, your task specification could be imperative, then planning/behavior orchestration declarative, then behavior implementation imperative again.
- **How important is a tight ROS integration for you?**
    - Francisco: If developing a deliberation library, it should be independent, but if you're building a framework for ROS, then you should design together with ROS.
    - Christian: Deliberation is historically under-represented in this space. The purpose of this group is to better represent ROS tools.
        - Sebastian: Originally it might have just been ROSPlan, smach, and… that's it?
    - Christian: Would have preferred that BT.CPP would have been way more ROS native, i.e., if BehaviorTree.ROS2 would have been more mature, or had more integrated rosbag logging, etc.
    - Daniel: Important to have a well-maintained library that supports the latest version(s) of ROS. Can sometimes get locked into older releases.
        - Christian: Also how well it interfaces with tools like ros2_control, moveit, nav2, etc.
- What kinds of visualization/debugging tools do you think are important for adoption of a deliberation tool?
- How do your answers change if you're doing a prototype vs. a product?

26-03-02

Technical
Jennifer Buehler will present a proposal for improvements to PyTrees!

- Ports w/ type checking + XML parsing support like BehaviorTree.CPP
- Proposal: Implement as a mixin on top of PyTrees Behaviour so it's opt-in

Discussion (slides)
Open discussion on Deliberation software design: interpreted vs. system languages, declarative vs. imperative, ROS integrated vs. optional, etc.

~~26-02-02~~

No meeting this month

26-01-12

Tool spotlight
Miguel Gonzalez Santamarta: YASMIN State Machines

- Repository: https://github.com/uleroboticsgroup/yasmin
- Demos: https://github.com/mgonzs13/yasmin_pyrobosim
- Presentation: https://github.com/uleroboticsgroup/yasmin/blob/main/docs/YASMIN-DeliberativeWG-2026-01-12.pdf

Discussion
Suggestions welcome!

25-12-01

Tool spotlight
Moritz Schmidt: soar_ros: A ROS 2 Interface for the Cognitive Architecture Soar

- Repository: https://github.com/THA-Embedded-Systems-Lab/soar_ros
- Soar: https://soar.eecs.umich.edu
- THA Robotics Research project "Cobot for my crafts": https://www.tha.de/Forschungsschwerpunkte/KI-Produktionsnetzwerk/Cobots-fuer-mein-Handwerk.html

Discussion
ROSCon x3 Deliberation Recap!

- ROSCon (Global)

    - RL workshop: https://github.com/ros-wg-delib/rl_deliberation (Christian + Sebastian)
    - https://plansys2.github.io/ (Francisco Martin)
- ROSCon ES
    - https://github.com/Eurecat/BTWorkshop_ROSCON_ES25/ (Devis dal Moro + Davide Faconti)
    - https://github.com/mgonzs13/yasmin_pyrobosim (Miguel Gonzalez Santamarta)
        - Videos: https://x.com/miggsant/status/1990366179555250564
- ROSCon DE/FR
    - soar_ros (see above)

Organizational

- Next meeting pushed 1 week out to 01/12/2026, to better accommodate New Year's vacation times.
- Who would be interested in contributing a technology to another ROSCon workshop in the style of 2024, where we demonstrated different technologies to solve ROS deliberation (https://github.com/ros-wg-delib/examples)?

~~25-11-03~~

No meeting this month, since we all expect to be recovering from ROSCon. 🙂

25-10-06

Tool spotlight (Marco Pastorio) – Simplified BT visualization

- Implemented in Groot1 because it's open-source, since Groot2 is closed
- Considering PyTrees, but concerned about high-frequency capabilities in Python
- Christian started a framework (https://github.com/boschresearch/bt_tools), works with BT.CPP and is purely intended for visualization purposes

Preview of ROSCon 2025 Reinforcement Learning Workshop (Christian + Sebastian)

- https://github.com/ros-wg-delib/rl_deliberation
- Kimberly: Make sure you give the right preliminary information introducing RL
    - +1 David
- Michele: How to choose action/observation spaces, and reward functions? Want validation that "I'm doing the right thing"
- Marco: Safety in RL systems, relationship with deliberation
    - Sebastian: Part of it is ensuring the action space / rewards prevent or penalize safety violations, but this doesn't provide guarantees. But also, having an outer safety/deliberation layer is necessary.
    - Marco: Could also train specific "recovery policies" that allow making contact and getting back to a recovery state.
    - Christian: Underlying planners/safety systems should ensure execution is safe

25-09-01

Tool spotlight: Pixi and robostack - a lightweight alternative to docker

- https://pixi.sh/latest/
- https://robostack.github.io/GettingStarted.html
- https://jafarabdi.github.io/blog/2025/ros2-pixi-dev/

Discussion: One the interface between deliberation and skill layer

- How can the border between an architectural skill layer and a deliberation layer be defined practically?
- What are the implications by the used deliberation tools?
- Should there be implications by the deliberation tools?
- Are the definitions different when using AI-tools on the deliberation layer vs classical methods like PDDL or BTs?

Christian: Intro Skill-based architectures:
https://www.sciencedirect.com/science/article/abs/pii/S092188902200207X

Guglielmo: Based on ROS Action

Deliberation layer

- Symbolic (e.g. grab a beer)

Skill layer

- Grab(object)
- (beer -> 0.3, 0.12, -.9)
- This could be application for a VLA layer

Missing in ROS atm: A way to manage symbols and their representation, i.e. world knowledge
E.g. by a semantic scene graph

http://www.open-ease.org/papers/beetz18knowrob.pdf
Guglielmo: “Robot matches video game in their head with the real world”

Francisco: Crucial is the way the robot predict what an action will change in the world.

Sebastian: A behavior tree action assumes the actions post condition. But if e.g. the action fails, things change.

Some learning methods on the skill layer may even require less symbolic knowledge on the deliberation layer

25-08-04

Announcing ROSCon 2025 deliberation workshop + talks
(talk acceptance notifications are on July 14)

Tool spotlight (ElSayed ElSheikh): Robot pick and place with behavior trees, NVIDIA Isaac Sim / cuRobo, and ROS 2

- Grab2 repo: [TODO]
- https://docs.google.com/presentation/d/1FZSjF_droZOfL-VyekMeBEHQMv4TF-ECzTzSWZuXdLE/edit?slide=id.gc6f80d1ff_0_66#slide=id.gc6f80d1ff_0_66

Continued group discussion: Generative AI for High-Level Behavior Orchestration

- Guglielmo: “PDDL MPC”
    - PDDL problem description
    - Mapping from PDDL symbolic action to subtree
    - Mapping from sensing data to symbolic predicates for PDDL
    - All actions are cancellable / abortable
    - Key issue: Mappings require *a lot* of engineering (i.e., an ontology)!
- Christian: “Expert models”
    - Language can be a middle ground
    - Example: Language → PDDL (or other formal model) → actions
    - How to map the language to the symbols in such a formal representation?
        - e.g., LLM/VLM to generate a valid PDDL problem?
- Agents (:rocket-emoji:)
    - LLMS being called repeatedly.
    - We can ignore this in robotics for now and focus on the chain from human input to (possibly complex) robot action in queries
    - Breaks free of having to engineer the grounding from state/actions to symbols
- Issues:
    - Assumption that all the information comes from perception, but in actuality you have to interact with objects (e.g., to extract mass properties, compliance)
        - More types of sensors (force/torque, tactile, etc.) can be useful
        - Still need to know when/whether you can take “exploratory” actions; how to do this without manually baking into the robot behavior?
        - We tend to learn skills and parameters, i.e., it’s easy for us to apply concepts across different skills and object types.
            - Example: If you learned to slice an apple, and that a pear is also a fruit, you can “zero shot” slice a pear.
            - We also are able to split concepts via experience, e.g., if you had a single pick skill, then maybe something will prompt you to do different skills (pick with 2 fingers vs. pick with full hand)
    - We don’t always deliberate everything up front; we do things reactively
        - How does this map to robot software?

    - Christian: And at different layers different depths of additional deliberation are needed (compare Attention Is All You Need)
    - We can reason in terms of goals – is the object picked?
        - We have a set of rich sensors that tell us whether our goal has been met
        - Equivalent to a “reward function” in RL
    - Reasoning is hierarchical (a loop of sense-plan-act)
        - Task planner -> BT -> navigation -> control
    - Demonstration (via language or teleop, or other) → exploration (reinforcement learning)
        - The human needs to also provide a reward system for whether a task is done, which is useful for both RL but also communicating back
        - Interpretability: How does a learned system convey why/how a skill succeeded or failed?
    - Christian: Here also the dialogical style or chatbots are interested in the theme of users interacting with the robot back and forth to get the object they wanted which implicitly creates the right label (compare On the origin of the hierarchy of color names)
    - Christian: From a SW-development perspective It boils down to the 'When all you have is a hammer everything looks like a nail'-problem in engineering and I think we are still lacking a nice understanding of what are the right problems to be fixed by LLMs, BTs, Motion planning or PDDL planning, etc respectively.
    - Sayed: Language models have been good at training visuomotor policies.
        - Use LLMs / VLMs to basically generate more synthetic data
        - Can help generalize with less data

- Other ideas:
    - Marco Pastorio: If anyone has some time to spare I'd love some feedback on the tutorial for ManyMove. I think I'm using BTs in a peculiar way, so insults are welcome too :D
    - https://github.com/pastoriomarco/manymove/blob/humble/manymove_cpp_trees/tutorials/tutorial_01.md
    - Daniel Yousef: Maybe the concept of a knowledge graph might be interesting here. That's how LLMs look up new world information. So maybe building a "knowledge graph"-like database might be a thing for sensing world data and saving it for later.

25-07-07

Announcement: New Deliberation community group category in Open Robotics Discourse: https://discourse.ros.org/c/community-groups/deliberation/127

David Conner: FlexBE behavior synthesis

- Synthesis from LTL specification to automaton

- Slides

Guglielmo Gemignani: Discussion – Generative AI for High-Level Behavior Orchestration
- Document outlining questions

- Related topic: Constructivism in AI

25-06-02

Florian Mirus, Frederik Pasch: Scenario Execution
- https://github.com/cps-test-lab/scenario-execution
- ROSCon presentation: https://vimeo.com/1024971964

- Not many tools for closed-loop testing exist
- Uses OpenSCENARIO
- Including behavior of the environment defined by behavior trees

David Conner: FlexBE + ROS 2 updates
- Christopher Newport University (CNU) maintaining the ROS 2 interface to FlexBE since ~2022
- TurtleSim demo: https://github.com/FlexBE/flexbe_turtlesim_demo
- Version 3 will remain the standard for Jazzy and older
- New FlexBE version 4 coming out, binaries to be available in ROS 2 Kilted this summer
- Reported high CPU usage in ROS 2 due to rclpy executors, should try events executor

25-05-05

Survey: Benefits and challenges for adopting behavior trees:
- https://discourse.ros.org/t/understanding-the-benefits-and-challenges-for-adopting-behavior-trees/43226
- https://forms.gle/3pJ6an5dxZXpzUpy9
- Similar results from deliberation group
    - https://zenodo.org/records/13382222
    - https://zenodo.org/records/14051492

Deliberation tool spotlight: TreeSense
https://discourse.ros.org/t/new-tool-for-working-with-behavior-trees/43369/4
- Repo: https://github.com/siddux/treesense

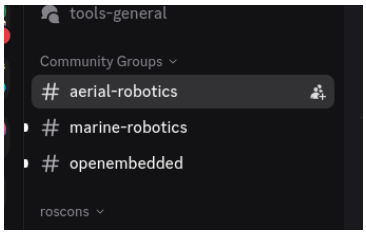
Discussion: Machine Learning policies in the ROS 2 deliberation stack
- Sebastian's slides: https://docs.google.com/presentation/d/1SHHniEJ_OJU-u-w9ku6H3rhn6N5D9-_8rzGK9beax-g/edit#slide=id.p

- Motivating example: https://github.com/FilipeAlmeidaFEUP/ros2_flatland_rl_tutorial
- Example from Louis Le Lay: https://github.com/louislelay/kinova_isaaclab_sim2real

25-04-07

Please provide feedback: https://forms.office.com/e/E6L8TSygpX

Proposal: Should we close https://discord.gg/Qt9veJ6WJx and move to a channel in the Open Robotics discord?



Deliberation tool spotlight: PyRoboSim
- Python API to create 2D-world
- Then provides ROS interfaces
- Main purpose: Task and Motion Planning
- Demo with PDDLStream: Usage/TAMP

European Robotics Forum (ERF) recap
Workshop: Advancing AI-Powered Robotic Cognition, Deliberation and Learning for Real-World Applications 250407_erf_ws_summary.pdf
- Discussion: When/how to use learning in robotics?
- David O: Summary of VLM for task planning as a FZI demo at ERF [Video link]
    - Uses detections from YOLO/SAM + a language prompt to generate behavior trees
    - Open question: How to prevent hallucinations in edge cases? Some model-based validation?
- Sebastian: UW colloquium talk

25-03-03

Deliberation tool spotlight: BehaviorTree.CPP / PyTrees / TurtleBot behavior demos
bt.cpp /
- skill definition in cpp
- XML for tree
- https://github.com/BehaviorTree/BehaviorTree.ros2 <- tree server
https://github.com/splintered-reality/py_trees
- ca 10 years old
- Probably less maintained

- BTs defined directly in python
ros2_ros_bt_py
JdeRobot BT Studio

Behavior Tree interpretability survey from Örebro University [link]
Earlier work https://www.sciencedirect.com/science/article/pii/S0921889024000976
Goal: researching what to influence
Standard claim of FSM being easier to read than BTs ..

Michele: It's also unclear, e.g. if people use reactive or non-reactive fallback etc.
Davide: BTs are a DSL and I can use any language to write something hard to read. But the question is how you use the language.

Sebastian Castro: I feel like it's almost more important for your SW to be approachable for beginners

Marco Pastorio:
https://discourse.ros.org/t/manymove-c-python-behavior-trees-for-robot-manipulators-looking-for-feedback-and-tips/42210

25-02-03

Please have a look our workshop repo and give feedback to us if and how it works as a resource for self-guided learning https://github.com/ros-wg-delib/roscon24-workshop/pull/85

Sebastian: Spotlight: Unified Planning:

- https://github.com/aiplan4eu/unified-planning
- https://github.com/sea-bass/learn-task-planning
- Samuele Sandrini: If anyone is interested in using the unified planning library with plansys2, here is a solver plugin for plansys2 that makes it possible to use unified planning engines as a solver (planner solver plugin of plansys2) https://github.com/JRL-CARI-CNR-UNIBS/plansys2_upf_plugin
- Christian: Compare https://github.com/RobotLabLTH/skiros2
- Andrea Micheli: Another repo I can recommend is the "embedded systems bridge": https://github.com/aiplan4eu/embedded-systems-bridge/
- https://github.com/PlanSys2/UPF4ROS2
- ROS1 wrapper: https://github.com/aiplan4eu/UP4ROS

Christian: I am involved in organizing an ERF workshop, where Francisco will talk about PlanSys2: https://lar-unibo.github.io/erf2025_ws42/

Discussion topic: Learning for Deliberation

- Sebastian: Presentation + link: https://docs.google.com/presentation/d/1BOtyzZqUwxLhAOZKQFkgN_phx6Jhr4Hsm7AgIkNd5rg/edit?usp=sharing
- Guglielmo Gemignani: https://github.com/nasa-jpl/rosa?tab=readme-ov-file
- Sachin Kumar https://github.com/sachinkum0009/nav2gpt

~~25-01-06~~

- Skipped because of the holiday period

24-12-02

- Discussion publication of workshop survey results
- Discussion special issue RA-P
- Maintenance topics of workshop repo

24-11-04

- Review of workshop results
    - https://survey.zohopublic.eu/zs/report/8CCSKc
- Presentation of new co-moderator
- New: meetings now on google meet
- Meetings in the future will not be recorded
- Discussion of new direction for community group
    - David Conner: interested
    - David Oberacker: interested
    - Integration with other projects like moveit or nav2
- Survey paper out of survey results?
    - Correlation of familiarity with *everything*
    - https://rose-workshops.github.io/rose2025/
    - https://www.ieee-ras.org/publications/ra-p

    - https://joss.theoj.org/
    - https://www.ieee-ras.org/publications/ra-l

24-08-05

(maybe we skip this meeting, because it is in our summer break)

24-07-01

- Preparation ROSCon Workshop

~~24-06-03~~

Skipped

24-05-06

Recording https://youtu.be/u9b1hJc5JbA

- Please note: This is now a **Community Group**
- **Task Manager**
  Taneli Korhonen, Karelics
  slides
    - https://github.com/Karelics/task_manager
  Notes
    - Tasks from different sources
    - Conflicting Tasks
    - Also unstable network

    - Single Interface to orchestrate tasks
    - By Voice / UI / CLI / Teleop
    - Monitor them
    - Mutually exclusive
    - List of tasks
    - Global Stop task

    - Can also feedback their state
    - Monitor active state

    - Missions combine multiple tasks in one
    - Overall state
    - Connection only required at the beginning

    - Cancellations can also come from somewhere else
    - (compare ROS actions)
  Questions
    - Focus on construction robots?
        - no
    - Should customers also be allowed to create own tasks?
        - generally, yes
    - Error management / recovery?

        - Currently not
        - But Planned
    - Matthias: Plans for Fallback etc
        - Not right now but probably sth like BTs
    - Enrico: Multi Robot vs Single Robot?
        - Currently only single robot
    - Calvin: Could you not make the blocking based on resources
        - We consider this

24-04-08

Recording https://youtu.be/UDXzRp7YQOA

- **ros2_ros_bt_py**
  David Oberacker, FZI Research Center for Information Technology
    - *oberacker@fzi.de*
    - https://github.com/fzi-forschungszentrum-informatik/ros2_ros_bt_py
    - Presentation at ROSConDE 2023 (in German)
- Discussion about joint workshop for ROSCon 2024 in Odense

24-03-11

Recording https://youtu.be/oLtowUhbPkU
No Agenda today
Open Discussion / General round of updates

- Is there a limit of how big data in the blackboard can be?
- How can subtrees be made reusable?
- How to execute BTs in ROS?
- Verification and Validation of Robotics Applications using BTs

~~24-03-04~~

canceled

24-02-05

Recording https://youtu.be/tfMqn5GWn9Q

- **A tooled approach to programming and execution of skill-based robotic behaviors**
  Matteo Morelli, CEA List
  slides
  - SW: squidly
  - Task:
    - Vision, Robot arm
    - assembly task
  - Architecture to orchestrate the skills

- Verification in autonomous driving https://leaderboard.carla.org/
- atomic skills orchestrated into more complex skills
- blue cylinders - shared knowledge base
- standalone runtime for BTs
- sl8 Davide: additional arguments rather than blackboard
- pull/push semantic to know which bt to execute
- Based on given state of the environment, which is the best policy to execute

- Discussion about Behavior Tree REP proposal
  https://github.com/ros-wg-delib/rep/blob/master/rep-2018.rst

24-01-01

NO MEETING

23-12-04

Recording https://youtu.be/dM8kSyTsokg

- **Deliberation in OpenRMF**
  M Grey, Intrinsic
  Parameters
  - Warehouse domain currently no considered
  - Multiple vendors of robots
  - Not all robots have the same interface
  - Centralized command is usually not an option
    - One fleet can have its own fleet manager
  - Shared schedule
  - Negotiation to resolve conflicts
    - Similar to CBS
  - Where happens deliberation
    - Coordination
    - Task allocation
  - “Elevators are a precious resource”
  - Every fleet demanded full control over a given lift-lobby
  - Traffic Negotiation (coordination)
    - Every agent gives its optimal path
    - Other agents propose how to avoid that
    - (multiple rounds)
    - Evaluation of costs
    - Execute best option
  - Task allocation
    - Predictive model to find individual bid
    - Runtime that executes this task plan
    - Current scope: single-agent-single-task (according to Gerkey)

        - Task Description JSON
            - Schemas per action
        - Bidding
            - New task is new auction
            - Bid is based also on previously assigned tasks which may be rearranged
              Cost is “sum of costs”
- Organizational
  <u>We will skip the meeting in January</u>
  Next meeting is 24-02-05

23-11-06

Recording: https://youtu.be/4aPd3WLWzCI

- Review ROScon WG Meetup 23-10-20
- ~~**The deliberation layer of ANYmal**~~
  ~~*Enea Scioni, ANYbotics*~~
- **Petri Nets for Robotic Deliberation**
  Thomas Horstink, Mainblades
  Thomas’ slides: ros_delib.pdf
  The library: https://github.com/thorstink/Symmetri

23-10-20

Meetup at ROSCon New Orleans

- Thomas Horstink
    - Presentation on petri nets
- Michael Grey
    - Open RMF
    - Presentation in Dec 4 meeting
- Francisco: Coresense

    - ROS-ification
    - Cognitive architecture
    - A metric that could also have semantic information
- Christian:
    - Convince
    - Formal verification in Robotics
- Davide
    - Wants to write things down
- Group: Commitment to review what Davide wrote

Presentation at ROSCon 2023

Supporting Robotic Deliberation: The Deliberation Working Group and Tools for Behavior Trees
Christian Henkel, Bosch
Video: https://vimeo.com/879001877/4978646728
Slides:
https://roscon.ros.org/talks/Supporting_Robotic_Deliberation_The_Deliberation_Working_Group_and_Tools_for_Behavior_Trees.pdf

23-10-02

Recording: https://youtu.be/lrb6-A2VboA

- Outlook ROSCon
    - I have booked Bird of the Feather Session in the Coffeebreak before my talk
    - I will send an E-Mail
- Has anyone worked with https://github.com/JdeRobot/BehaviorTrees ? Can you report?
    - Francisco: Is part of my university
    - Have recently hated ROS
    - Code is in very basic state
    - No documentation
    - Generates pytrees implementation out of bt xml files
    - https://github.com/BehaviorTree/BehaviorTree.CPP/pull/634
- Discussion about the potential of standardizing naming of behavior tree control nodes and of action definitions
    - My slides:
      # Motivation I
      - Currently it is unclear what some control nodes do
          - Mapping between name and concrete behavior in all cases
          - Behavior on `halt` signal
      - How should _skills_ that the BT interact with be structured?
          - What if they use the same (physical) resource

Motivation II
- Necessity for formal verification i.e. model checking
- Static analysis of a BT, e.g. detection of deadlocks, unreachable code

---

Suggested Steps I
- Define BT...
    - ... control nodes
    - ... decorators
    - ... leaf nodes
    - ... tick behavior / propagation
- Should be testable against a given implementation

Suggested Steps II
- Define
    - Guidelines what actions and conditions should can or cannot do to the system
        - e.g. A condition must always return within a $t \ll t_{\texttt{tick}}$

First steps
- I prepared a REP
    - https://github.com/ros-wg-delib/rep/blob/master/rep-2018.rst
- Please have a look
    - Send me your github user via E-Mail if you want to join our organization

- Jamie: Sysml has similar problems
    - Ancestor of UML
    - Also does deadlock, reachability
    - Can specify state machine
- Recap of discussion with Davide
    - @Davide could you please add your slides, here?
- Michele:
    - Behavior on resetting of nodes
- Sebastian:
    - Also flag to user at design time, for example if he makes a parallel node with 200 children, which would create a timeout even with short runtimes per tick
- Jamie:
    - What about flowstate (https://intrinsic.ai/blog/posts/introducing-intrinsic-flowstate/) ?
    - We should include them here
- Christian:
    - What is a good way to define the the condition nodes that is both readable and executable?
- Sebastian:
    - Maybe just rely on the xml for bt.cpp

    - Enrico:
        - SCOPE had a formal definition of some control nodes
        - Armando worked on these
        - Suggestion: A given BT should produce a precise sequence of ticks

23-08-07

Recording: https://www.youtube.com/watch?v=UIsGCSJAP7M

- Organizational:
    - No WG meeting in September
    - Next meeting 23-10-02
- **Deliberation at NODE - Quirks and Features**
  Jannik Abbenseth, NODE Robotics
    - "One does not simply run rviz in production"
    - "This is where we start"
    - mbf_state_machine - move base flex state-machine
        - Kind of a state-machine that selects recovery behaviors based on the given situation.
        - Downside: Parallel execution is hard, introspection is hard
    - BT use inspired by nav2 using it
        - Problems caused by using ROS1
        - Missions are json serialized
        - Dynamically build and extend a tree
        - Structures of json and tree are related
        - Decorators own children, vs one root node owning everything
        - Groot does not work anymore, because trees are automatically built
    - Some costumers don't have Warehouse Management Systems (WMS)
    - So this may also be required
    - "Robotics is hard"
    - Expandable Sequence is a special BT control node, where you can add children at runtime, is running if no child exists
- **Continuation of discussion on different behavior tree formalism**
  Enrico Ghiorzi, Università di Genova
    Questions on halting:
    - How to implement?
    - In what order?
    - Why call halting if child was not even executed?
    - What if halting takes time?

23-07-03

Recording (audio only): https://youtu.be/mSQYPHIwKK0

- Welcome

- **SkiROS2: A Skill-based Robot Control Platform for ROS**
  Matthias Mayr, Lund University & Francesco Rovida, RiACT
  SkiROS2 is a skill-based robot control platform with a layered, hybrid control structure for automated task planning, and reactive execution, supported by a knowledge base for reasoning about the world state and entities. The scheduling formulation builds on the extended behavior tree model that merges task-level planning and execution. The skill formulation based on pre-, hold- and post-conditions allows to organize robot programs and to compose diverse skills reaching from perception to low-level control and the incorporation of external tools. SkiROS2 is written in Python, integrates into ROS and is available as open-source software at https://github.com/RVMI/skiros2.
- **Observations of differences between behavior tree formalism and implementation**
  Enrico Ghiorzi, Università di Genova
  Link to slides

From discussion:

- Maybe undertake joint effort to formulate a standard, namely a REP, for defining semantics of control node types - together with main developers of relevant libraries.
- Preemption is an open problem with BTs in real robotic systems - in contrast to computer games, where actions can always be stopped instantaneously.

23-06-05

Recording: https://youtu.be/x8stFrsiVco

- Organizational
  - Recap last meeting
  - Now regular meeting every first Monday 5pm
  - Github Organization
    https://github.com/ros-wg-delib
  - Awesome list created for collection of topics
    https://github.com/ros-wg-delib/awesome-ros-deliberation
  - Youtube Channel
    https://www.youtube.com/@ros-wg-deliberation
  - Christian will be in a train to Bordeaux next meeting. Ralph will help out with moderation.
  - If you are also at ROS Industrial Conference, I would be happy to chat
- **BehaviorTree.CPP: what makes it different from other implementations**
  Davide Faconti, PickNik Robotics
  - Link to slides
  - Not all behavior tree implementations are created equal
  - Issue with model-driven development: often requires the understanding of code and model
  - In BT.CPP: Model should provide sufficient information to understand system
  - Vs https://pypi.org/project/py-trees/: Tree definition is in one place and directly understandable
  - Dataflow: other implementations make it too opaque

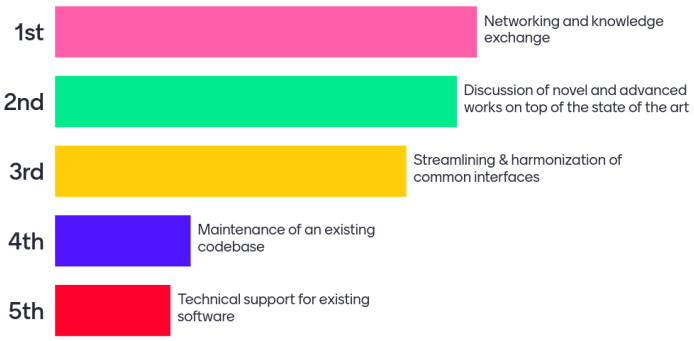
    - Blackboard is part of the model / exposed to the user
    - Problem of global variables is true in composition
    - New in BT.CPP 4: Pre- and post-conditions with custom scripting language
    - Questions:
        - Can the BT be modified online?
            - Not during execution without stopping the tree
            - Problem: Stateful nodes
            - Possible without recompiling but state is lost
        - Can the state be snap-shoted?
            - At the moment not, but would be great
        - Is the execution of the actual action done in another node?
            - Yes, this is best practice
        - Have you thought about standardizing the interfaces?
            - Standardizing by using is the best option (not standardizing by committee)
            - Written guideline would be nice to have
        - Roadmap for Groot2 features?
            - Commercial SW
            - Free for academia
            - Partially for industry
            - Break-points are implemented already (companies are testing it, official release will come soon)
        - Wouldn't it make sense to use more standard nodes?
            - No, use it as a domain specific language
            - I try to add useful patterns but not everything people request
- **Integrated Task and Motion Planning with ROS 2 and PDDLStream**
  Sebastian Castro, PickNik Robotics
    - Link to slides
    - Pipeline from goal specification, PDDL planning to behavior trees for execution.
    - Demo of https://github.com/sea-bass/pyrobosim
    - Task planning vs Task and motion planning
    - TAMP precomputes motions to take into account for planning
    - Can be viewed as partial automation of domain design
    - Tool that does this is https://github.com/caelan/pddlstream
    - Some parameters of pddl have python code injected that must be evaluated at planning time
    - For example to check feasibility
    - Standardization:
        - A lot of potential
        - Extension of plansys2 to allow for TAMP
        - Stop reimplementing the BT execution
    - Questions:
        - How can the actions be more coupled to the tree?
            - Plansys2 does this to a degree

- Closing
    - Who wants to present next time?

23-04-28

Recording: 2023-04-28 First Deliberation WG Meeting

- Welcome
- What is Robot Deliberation? Ralph Lange, Bosch Research (10')
    - Other terms like mission layer, executive layer, etc.
    - Within robotics sw architecture
    - Also contingency handling
    - Deliberation technologies that exist, theoretically
    - Planning based example, e.g. shakey
    - Big number of DSLs show that this is unsolved
    - Also in ROS there are many existing packages that do deliberation
    - Introduction to convince and it's use cases
    - Approach and objectives of convince
- Motivation of WG, first goal



-
    - First working goal is a subpage on docs.ros.org with a collection of existing packages that do deliberation
- Using Formal Methods to Synthesize and Verify Behavior Trees, Jamie Smith, Imandra (15')
    - Formal verification of controller SW
    - Automated discovery of system behavior
    - Scenario synthesis for simulation-based testing
    - Mission plan verification
    - Create tooling for formal semantics for BTs
    - Verify BT against formal properties
    - Use region decomposition to explore state space
    - Questions about flow and execution
    - Real-time warnings: background verification
    - Scale to multi-robot scenarios
    - Q&A

        - Which part of the system are you aiming at? → BT layer (very interesting, top candidate) or mission planning layer. But also having solutions for control layer - currently explored with individual customers.
        - On which application scenarios do you focus at Imandra? → mobile robots, collaborative robots, logistics robots (with humans around), medical robots, in general robots in open environments with other autonomous agents.
- Deliberate and Act with PlanSys2, Francisco Martín Rico, Universidad Rey Juan Carlos (15')
    - slides: CodingBehaviors_WG_Deliberation.pptx
    - Project with axiom company with mobile robots in an environment shared with humans
    - Formalisms in use: FSMs and BTs
    - ROS 2 Planning System
    - Command line tooling to specify goals for planner and to query world model
    - Integration and use of UPF (Unified Planning Framework) from AIPlan4EU, see https://www.ai4europe.eu/research/ai-catalog/unified-planning-framework.
    - Cognitive architecture: Knowledge graph used to represent the environment in symbolic fashion.
    - Q&A
        - How to trigger direct actions vs BTs? → Generic PlanSys2 action type with parameters for ROS 2 Actions and BTs. All are triggered via the *Action Hub*, implemented by a single topic /action_hub.
        - Multi-robot scenario: how to model different skills of robots/ different goals: parameters in actions. The action performers can be configured to attend only to the action execution requests that are equals to a preconfigured parameter. You can include in the actions a parameter that indicates the robot, and each robot parametrize its actions with its id.
        - Instead of PlanSys2 orchestrating execution of a plan, can it return a standalone plan in some format (e.g. BT) that can be passed into another software module to manage execution? → yes, should be possible
        - Cognitive architecture: do you use some learning techniques: no, but using ontologies is the next step
        - Does the planner construct the BTs dynamically: AMAS paper (https://www.ifaamas.org/Proceedings/aamas2021/pdfs/p1596.pdf), creates static BT from planning graph.
- AIPlan4EU and Planning with Behavior Trees, Guglielmo Gemignani, Magazino GmbH (10'):
    - slides: https://docs.google.com/presentation/d/17i0DXUC89jJq-FUYytRZLFvMtqbU-CLkSOARitrK6YE/edit?usp=sharing
    - Unified planning framework (UPF): library to seamlessly invoke a portfolio of planning techniques: https://github.com/aiplan4eu/unified-planning
    - Wrapper for ROS 1 https://github.com/aiplan4eu/UP4ROS

- Wrapper for ROS 2 in work, cf. https://github.com/aiplan4eu/UP4ROS2/tree/repo_migration.
- Interfacing with PlanSys2, using PDDL
- ICAPS 2023 paper: Planning for automated testing of implicit constraint behavior trees: https://g-gemignani.github.io/files/papers/Koeckemann-etal-2023-ICAPS.pdf
- Q&A:
    - Is there an easy, documented way to import BTs: yes, JSON-flies (not in the paper, please contact directly)

- Your Input!

- Methods for robotic deliberation:
    - Classical planning
    - Planning and acting
    - BTs
    - Htn
    - Hfsm
    - Symbolic AI
    - Petri nets
    - Formal semantics
    - Bdi agents
    - Merlin 2
    - PlanSys2
    - Pddl
    - Yasmin
    - Cognitive architectures
    - Reasonsers
    - Knowledge
    - Prolog
    - Json

        - Maps-k
        - Knowrob
        - Acumos
        - State machines
        - Bdl
        - flexbe
- Organizational
    - Move to 5 pm to make it accessible for US
    - 1 hour duration
    - Day: Monday
- Actions items:
    - Talk on generic interface for integrated task and motion planning (ITAMP) by Sebastian Castro, cf. https://github.com/sea-bass/pyrobosim.
        - Title: Integrated Task and Motion Planning with ROS 2 and PDDLStream
    - Presentation (at the next meeting) about BehaviorTree.CPP and its novel ideas, with respect to other implementations, from Davide Faconti