SPACE ROS: ARCHITECTURE ROADMAP

Contributors:

Paul Tompkins, Blue Origin
Austin Probe, NASA
Amalaye Oyake, Blue Origin
Will Chambers
Overview

- Statement of Objectives
- Driving Reference Applications
- Technology Need
  - NASA Taxonomy
  - Community Input
- Deployment Configurations
  - Middleware Solutions
  - Space ROS Deployment
- Benchmarking
  - Benchmarking Objectives
  - Test Matrix Summary
- Space ROS Development
  - Governance
  - License
  - Pull Request Acceptance
  - Community Guidelines
Statement of Objectives

- Develop a space-certifiable and reusable robotic framework
  - Designed to meet flight software standards: Designed to be aligned with NASA so that it could be adopted for a Class A Mission
    - Memory Safety
    - Deterministic Performance
    - Static analysis
    - System Testing
  - Support characteristic space robotics applications
  - Enable rapid development of new robotic capabilities
  - Enable reuse of capabilities between missions
Toward a Common In-Space Command, Control, Communications and Compute Infrastructure

Human Controlled Robotics

Maintenance & Repair

Positioning Systems

Mining & Excavation

Spacecraft Autonomy

OSAM

Flying Robots

Future Robotic Systems

Future Surface Habitats

Distributed Edge Computing

Collaboration Networks & Teleoperation Autonomy

System/Network Control Framework

Flying Robots

Rovers

ISRU

Toward a Common In-Space Command, Control, Communications and Compute Infrastructure
Driving Reference Applications

- A set of reference applications act as a litmus test for:
  - Early assessment of existing (and missing) ROS features for space
  - Identifying the driving attributes of Space ROS to achieve “typical” / projected space robotic systems

- Objectives:
  - Establish a systematic approach to specifying space robotics applications that motivate the definition of essential Space ROS features
  - Identify a set of space robotics applications that collectively motivate the majority of features we hope Space ROS to embody

- This package:
  - Proposes an application spec template for uniform treatment of candidate applications
  - Provides preliminary specifications for a number of space robotics applications

- With a first round of feedback, we can expand concepts more thoroughly
Surface: Cargo Offloading

- **Description**
  - Robotic lift, translation and placement of cargo from lander deck to surface

- **Key Elements**
  - Robotic Crane
  - Lander
  - Payload(s)
  - Mission Control (Earth)

- **Technical Attributes**
  - Perception
    - Payload location/segmentation
    - Lander structure perception
    - Surface characterization
  - Planning
    - Manipulator trajectory planning
    - Payload surface placement planning
  - Manipulation
    - Payload grapple
    - Payload translation under gravity and dynamics
    - Payload surface placement amongst terrain features
  - Control and Coordination
    - Earth-based human supervision
    - Interleaved lander and robotics tasks (e.g. payload umbilical and mechanical release)
    - Possible payload health monitoring before/during/after crane operation

- **Communication**
  - Crane-Lander: hardline (serial, Ethernet)
  - Payload-Lander: hardline (via severable umbilical)
  - Lander-Mission Control: RF
  - EXTENSION: Payload-Crane (via grapple fixture?)
Surface: Resource Prospecting

• Description
  • Mobile survey to characterize regions for amenability for later resource extraction

• Key Elements:
  • Mobile robotic resource prospector
  • Instruments
  • Drill
  • Orbital (or surface) communications relay(s)
  • Mission Control (Earth)

• Technical Attributes
  • Perception
    • Natural terrain characterization (traversability) in low light (lunar)
    • Detection of suitable drive paths and sampling sites
    • Novel feature/signal detection
  • Planning
    • Traversal planning considering remote sensing (resource maps), lighting, communications
    • Activity planning considering battery state of charge, intended activities and resource costs
  • Mobility
    • Control under rover kinematics and surface terramechanics
    • Terrain obstacle avoidance
    • Instrument, manipulator and/or drill placement
  • Manipulation
    • Possible active regolith interaction (scooping, scraping)
    • Drilling / subsurface access
  • Control and Coordination
    • Earth-based human supervision
    • Rover, instrument, manipulator and/or drill coordinated control
  • Communications
    • Prospector-Instruments: hardline (serial, Ethernet)
    • Prospector-Drill: hardline (serial, Ethernet)
    • Prospector-Mission Control: RF (direct-to-Earth)
    • Prospector-Comm Relay: RF
    • Comm Relay-Mission Control: RF
Surface: Site Preparation

- **Description**
  - Terrain preparation and foundational construction in support of advanced surface operations, including:
    - Landing
    - Habitat emplacement
    - In-situ resource utilization

- **Key Elements**
  - Robotic Regolith Movers (robot team; TBD composition and roles)
  - Mission Control

- **Technical Attributes**
  - **Perception**
    - Site characterization, including terrain volumetric assessments
    - Terrain mechanical resistance, mass
    - Other robot detection
  - **Planning**
    - Robot team-based excavation, deposition and leveling planning
    - Local route planning
    - Activity planning considering battery state of charge, intended activities and resource costs
  - **Mobility**
    - Control under rover kinematics and significant surface interactions
    - Terrain obstacle avoidance
    - Tool / manipulator placement and control under motion
  - **Control and Coordination**
    - Robot team coordination (loose and potentially tight coordination)
    - Earth-based human supervision
  - **Communication**
    - Robot-Robot: RF / WiFi
    - Robot-Mission Control: RF
Surface: Orbital Station Unpressurized Robotics

- Description
  - External (unpressurized) service tasking in support of orbital space station operations

- Key Elements
  - Station Modules
  - Visiting Vehicles
  - Orbital Replacement Units
  - Mission Control
  - Crew Workstation

- Technical Attributes
  - Perception
    - Proprioception (self-configuration)
    - Obstacle detection (module hulls, visiting vehicles, external payloads and equipment, astronauts)
    - Force/moment detection (contact)
  - Planning
    - Arm trajectory planning
    - Task planning
  - Mobility
    - Control under arm kinematics, base contact and payload mass properties
    - Station obstacle avoidance
  - Manipulation
    - Closed-chain grappling/initial contact, ungrappling
    - Free-flyer grappling/initial contact, release
    - Maneuvering “payloads” under arm kinematics and joint payload/station/arm contact avoidance, force and torque limits
    - Emergency halt in face of unexpected contact or resistance

- Control and Coordination
  - Earth-based human supervision
  - Direct human control
  - Station GNC coordination (collaborative mode switching, motion control, fault response)
  - Visiting vehicle coordination (collaborative mode switching, motion control, fault response)

- Communications
  - Station-Manipulator: hardline
  - Station-Visiting Vehicle: RF and/or hardline (state-dependent)
  - Station-Crew Workstation: hardline
  - Station-ORU: hardline (via standard quick connect/disconnect interface)
  - OPTIONAL: ORU-Manipulator (for health monitoring/status)
  - Station-Mission Control: RF
Technology Need
This taxonomy identifies, organizes, and communicates technology areas relevant to advancing the agency’s mission.

Provides a structure for articulating the technology development disciplines needed to enable future space missions.

We identify several of the NASA Technology areas where Space ROS can have the most meaningful impact.

https://www.nasa.gov/offices/oct/taxonomy/index.html
NASA Technology Taxonomy - 04 Robotic Systems
NASA Technology Taxonomy - 07 Exploration Destination Systems
NASA Technology Taxonomy – 10 Autonomous Systems

TX10 Autonomous Systems

- **TX10.1 Situational and Self-Awareness**
  - 10.1.1 Sensing and Perception for Autonomous Systems
  - 10.1.2 State Estimation and Monitoring
  - 10.1.3 Knowledge and Model Building
  - 10.1.4 Hazard Assessment
  - 10.1.5 Event and Trend Identification
  - 10.1.6 Anomaly Detection

- **TX10.2 Reasoning and Acting**
  - 10.2.1 Mission Planning and Scheduling
  - 10.2.2 Activity and Resource Planning and Scheduling
  - 10.2.3 Motion Planning
  - 10.2.4 Execution and Control
  - 10.2.5 Fault Diagnosis and Prognosis
  - 10.2.6 Fault Response
  - 10.2.7 Learning and Adapting

- **TX10.3 Collaboration and Interaction**
  - 10.3.1 Joint Knowledge and Understanding
  - 10.3.2 Behavior and Intent Prediction
  - 10.3.3 Goal and Task Negotiation
  - 10.3.4 Operational Trust Building

- **TX10.4 Engineering and Integrity**
  - 10.4.1 Verification and Validation of Autonomous Systems
  - 10.4.2 Test and Evaluation of Autonomous Systems
  - 10.4.3 Operational Assurance of Autonomous Systems
  - 10.4.4 Modeling and Simulation of Autonomous Systems
  - 10.4.5 Architecture and Design of Autonomous Systems
NASA Technology Taxonomy – 11 Software, Modeling, Simulation, and Information Processing
Community Feedback

- NASA solicited feedback on the Space ROS concept from the space robotics and autonomy community
- Responses were received from 30 companies, universities, and individuals
RFI Takeaways: Statistics

Robotic Cardinality

Mode of Perception

Operational Environment

Mode of Locomotion/Mobility
RFI Takeaways: Statistics

### Degree and Mode of Manipulation
- **Single Arm**: 40%
- **Muti Arm**: 30%

### Control Topology
- **Central**: 50%
- **Distributed**: 40%
- **Hierarchical**: 10%

### Degree of Autonomy
- **Autonomous**: 60%
- **Teleoperation**: 30%
- **Full**: 10%
- **Semi**: 0%

### System Communication
- **Inter-Process**: 70%
- **Inter-Robot**: 60%
- **Ground**: 0%
RFI Takeaways: Statistics

System Communication

Processor Considerations

Sensors and Actuator Drivers

Sensor Data processors
RFI Takeaways: Statistics

Software/Framework Interoperability

- ROS2: 80%
- cFS: 10%

Algorithms

- Fusion: 70%
- Planning: 60%
- FDIR: 50%
- SLAM: 30%
- Machine Learning: 10%
RFI Takeaways: Space ROS

- There is significant interest in a version of ROS2 that is capable of supporting space systems and robotics
  - Operating in a resource constrained environment is critical
  - Interactions with embedded systems is a concern for many respondents
  - Security is important to organizations working with the DoD

- Documentation and examples are a key to adoption

- A significant portion of respondents are interested in supporting the development community
RFI Takeaways: ROS Packages

The RFI requested what ROS packages would be valuable to the community. The following packages were mentioned in two or more responses.

<table>
<thead>
<tr>
<th>Package</th>
<th>Votes</th>
<th>Package</th>
<th>Votes</th>
<th>Package</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>tf2</td>
<td>11</td>
<td>ament_index_cpp</td>
<td>2</td>
<td>Rclcpp</td>
<td>2</td>
</tr>
<tr>
<td>ros_control</td>
<td>6</td>
<td>rostopic</td>
<td>2</td>
<td>image_geometry</td>
<td>2</td>
</tr>
<tr>
<td>Movelt</td>
<td>6</td>
<td>Eigen</td>
<td>2</td>
<td>diagnostic_updater</td>
<td>2</td>
</tr>
<tr>
<td>RViz</td>
<td>5</td>
<td>interactive_markers</td>
<td>2</td>
<td>Rclcpp_action</td>
<td>2</td>
</tr>
<tr>
<td>urdf</td>
<td>5</td>
<td>angles</td>
<td>2</td>
<td>kdl</td>
<td>2</td>
</tr>
<tr>
<td>xacro</td>
<td>5</td>
<td>Pluginlib</td>
<td>2</td>
<td>gazebo_ros_pkgs</td>
<td>2</td>
</tr>
<tr>
<td>robot.localization</td>
<td>4</td>
<td>joint_state_publisher</td>
<td>2</td>
<td>gazebo_msgs</td>
<td>2</td>
</tr>
<tr>
<td>Gazebo</td>
<td>4</td>
<td>Rosidl_default_generators</td>
<td>2</td>
<td>geometry_msgs</td>
<td>2</td>
</tr>
<tr>
<td>image_transport</td>
<td>4</td>
<td>rtabmap</td>
<td>2</td>
<td>Nav_msgs</td>
<td>2</td>
</tr>
<tr>
<td>cv_bridge</td>
<td>3</td>
<td>behavior_tree</td>
<td>2</td>
<td>Visualization_msgs</td>
<td>2</td>
</tr>
<tr>
<td>Std_msgs</td>
<td>3</td>
<td>Builtin_interfaces</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sensor_msgs</td>
<td>3</td>
<td>rqt</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trajectory_msgs</td>
<td>3</td>
<td>trace_tools_test</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rosbag</td>
<td>2</td>
<td>PCL</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dynamic_reconfigure</td>
<td>2</td>
<td>Robot_state_publisher</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rcutils</td>
<td>2</td>
<td>camera_info_manager</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Architecture Concepts
Space ROS App Definition

- Executes one or more system functions
- Multiple communications interfaces
- Comprised of several packages
- Apps can leverage support package to provide more standardized FSW app implementation (template)

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node App</td>
<td>Space ROS App that communicates solely with the message Space ROS message bus</td>
</tr>
<tr>
<td>Interface App</td>
<td>Provides a Space ROS compliant interface to an embedded sensor/system. Can be used to provide an abstraction to the underlying system</td>
</tr>
</tbody>
</table>
| Bridge App       | Bridges Space ROS to other middleware implementations  
                     Could be mission agnostic and auto-generated |
Space ROS is a software framework designed to be deployed across a wide set of use cases.

There are a number of ways Space ROS could be deployed:

1. Standalone

2. Federated: Loosely coupled with another framework (such as cFS)

3. Integrate Modular Avionics (IMA): Tightly integrated within the same computing using with another framework
**Standalone vs. Federated vs. IMA**

- **Standalone**: A single unit that encapsulates the entire system functionality and uses Space ROS may increase mass because it is its own physical payload but will require Space ROS to meet the validation requirements for the mission independently.

- **Federated**: an independent unit encapsulates Space ROS communicates with the Avionics. This may increase mass because it is its own physical payload but reduces V&V from the flight software team.

- **IMA**: a unit that is tightly integrated with the underlying Flight Software will impose some V&V work on the flight software team, but may save mass, because the compute capability lives in the centralized flight software computer.
# Deployment Configuration Types

**Space ROS Deployment**

- Augmented System
- Single Vehicle
- Multi Vehicle

**Avionics**

- Integrated
- Shared
Augmented System Overview

Integrated Avionics

Shared Avionics

Bus Avionics

Bus Middleware
Apps

App 1

App 2

App N

Bridge
App

Space ROS System 1 Avionics

Space ROS Apps

App 1

App 2

App N

Space ROS System N Avionics

Space ROS Apps

App 1

App 2

App N

Bridge
App

Bus Avionics

Bus Middleware
Apps

App 1

App 2

App N

Space ROS System 1 Avionics

Space ROS Apps

App 1

App 2

App N

Bridge
App

Bus Avionics

Bus Middleware
Apps

App 1

App 2

App N

Space ROS System N Avionics

Space ROS Apps

App 1

App 2

App N

Bridge
App

Space ROS System N Avionics

Space ROS Apps

App 1

App 2

App N

Bridge
App
Augmented System App Architecture

- Bus App
  - Radio App
    - Radio
  - Bus Interface App
    - Sensor / System
- Payload Interface App
- Space ROS Bridge App
- Space ROS Node Apps
  - Space ROS Interface App
  - Sensors / Systems
- FSW Middleware Message Bus
  - Space ROS Message Bus
  - Ground Link
Single Vehicle System Overview

Integrated Avionics

Shared Avionics
Single Vehicle System App Architecture

Reusable, mission agnostic apps

Subsystems include
- Payloads
  - i.e. robot arm
- Bus Interfaces
  - i.e. Electrical Power System

Space ROS Subsystem

Space ROS Bus Node Apps

Space ROS Node Apps

Space ROS Interface App

Sensors / Systems

Space ROS Message Bus

Radio App

Radio
Multi Vehicle System Description

- Peer to Peer comms via RF link
- Use Cases
  - Heterogenous middleware environment
    - Intervehicle communications via Multi Agent Interface Language
    - Space ROS message domain does not extend beyond single vehicle
  - Space ROS only
    - Space ROS message domain extends beyond single vehicle
  - Mixed?
Multi Vehicle System Description

Intervehicle Comms via RF
Ground Comms

Spacecraft 1
No Space ROS Present

Spacecraft 2
Augmented System

Spacecraft N
Single Vehicle Space ROS

Bus Software Apps

Radio App

Ground Station

Ground Comms

Bus Software Apps

Radio App

Radio App

Space ROS Subsystem 1 Apps

Space ROS Subsystem N Apps

Radio App
Ground-only Space ROS System Overview

Ground Subsystem

Flight Subsystem

Ground Data System

Ground Software Apps

Telem and Cmd Server

Ground Station

Space ROS Ground Subsystem 1

Space ROS Apps

App 1

App 2

App N

Bridge App

Space ROS Ground Subsystem 2

Space ROS Apps

App 1

App 2

App N

Radio

Flight Subsystem

Middleware System N

Apps

Middleware System 1

Apps

Middleware System N

Apps

Radio

Ground Station
Shared Space ROS Ground and Flight System Overview

Ground Subsystem

- **Ground Data System**
  - Ground Software Apps
    - App 1
    - App 2
    - ... App N
  - Telem and Cmd Server

- **Space ROS Ground Subsystem 1**
  - Space ROS Apps
    - App 1
    - App 2
    - ... App N

- **Space ROS Ground Subsystem 2**
  - Space ROS Apps
    - App 1
    - App 2
    - ... App N

Flight Subsystem

- **Flight Subsystem**
  - Space ROS Bus Apps
    - App 1
    - App 2
    - ... App N

- **Space ROS System 1 Apps**
  - App 1
  - App 2
  - ... App N

- **Space ROS System N Apps**
  - App 1
  - App 2
  - ... App N

- **Radio**
  - Ground Station
Shared Ground and Flight System App Architecture

- Space ROS Ground System
  - Space ROS Node Apps
  - Space ROS Bus Node Apps
  - Space ROS Interface App
  - Data Stores / Servers

- Ground Software Message Bus
  - Ground Software Apps
  - Space ROS Bus Node Apps
  - Space ROS Bridge App

- Space ROS Message Bus
  - Tlm and Cmd Server
  - Space ROS Interface App

- Space ROS Flight System
  - Space ROS Node Apps
  - Space ROS Bus Node Apps
  - Sensors / Systems

- Reusable, mission agnostic apps

- Ground Station
Space ROS used for Testing and Simulation

Ground Subsystem

Flight Subsystem

Ground Data System

Flight System Testing and Simulation

Space ROS Apps

App 1
App 2
... App N

Ground System Testing and Simulation

Space ROS Apps

Telem and Cmd Server

Bridge App

Flight System Testing and Simulation

Space ROS Apps

App 1
App 2
... App N

Middleware System 1

Radio

Middleware System N

Radio

Ground Station
Space ROS Benchmarking
Benchmarking Objectives

We are working to evaluate the build of Space ROS under development to:

- Characterize Space ROS performance
  - Processor Utilization
  - Memory
  - Network throughput
  - QoS Metrics

- Compare Performance to other frameworks using Ames Distributed Space Autonomy Suite
  - Compare to NASA cFS with reimplemented applications for Autonomy and Navigation

- Space-like targets
  - X86 Architecture e.g. Unibap iX5-100
  - ARM Architecture e.g. ZCU-102
# Benchmarking Test Matrix Summary

A full test matrix has been developed for evaluating Space ROS in both a containerized development environment and in a processor-in-the-loop environment on target hardware including:

<table>
<thead>
<tr>
<th>Test Description</th>
<th>Metric Recorded</th>
<th>Metric Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message passing tests using but a fully connected and linear mesh network with: Ideal, 10% loss, 50% loss, and 90% loss, 100 ms latency and 500 ms latency network configurations</td>
<td>Deployed Application Size</td>
<td>DSA Apps + SpaceROS Compiled File Size</td>
</tr>
<tr>
<td></td>
<td>CPU Utilization</td>
<td>SpaceROS CPU Utilization, max, min, mean, variance</td>
</tr>
<tr>
<td></td>
<td>Memory Usage</td>
<td>SpaceROS Memory Utilization, max, min, mean, variance</td>
</tr>
<tr>
<td></td>
<td>Topic Latency</td>
<td>Latency between nodes reward and assignment messages, max, min, mean, variance</td>
</tr>
<tr>
<td></td>
<td>Inter Container Latency</td>
<td>Latency between AUTO nodes Auto Sat State Message</td>
</tr>
<tr>
<td></td>
<td>Inter Container Packet Loss</td>
<td>Number of AUTO Sat State Messages not received.</td>
</tr>
<tr>
<td></td>
<td>Total State Messages Ignored</td>
<td>When a message is delayed too long it is just ignored so this combined both dropped packets and ones whose latency was too large.</td>
</tr>
</tbody>
</table>
Space ROS Community
Space ROS Community Governance

- Space ROS will seek to leverage ROS
  - Space ROS trademarks being transferred to Open Robotics
  - Following ROS2 Governance Guidelines: https://docs.ros.org/en/galactic/Governance.html

- Space ROS will form a Technical Steering Committee (TSC) to prioritize feature selection
  - Member slots available in exchange for TBD FTE support (e.g. 0.5 FTE-1 FTE)
  - Membership is available to organizations and missions

- Space ROS will attend relevant community events to advertise and seek community input
  - e.g. ROS Con, FSW Workshop
Governance and Post ACO Operations

- Technical Steering Committee
  - Volunteers to identify priorities and features
  - Looking to identify members of government and industry groups to serve as members

- Project Manager x1
  - Interfaces with the community manages pull requests

- Testing and Continuous Integration Management x2
  - Maintains testing and V&V infrastructure
  - Supports maintaining compatibility with ROS2
Space ROS Public Working Group

- We are proposing a working group meeting separate from the ACO effort to integrate external partners into Space ROS development
  - Companies and organizations who have expressed interest in supporting development
  - Companies and organizations who responded to the RFI
  - Allow for Will Chambers to maintain involvement

- Open Questions?
  - Meeting cadence
  - How quickly to invite new people
Space ROS Community Pull Request Acceptance Policy

- TSC identifies pull requests for inclusion

- Pull requests must Space ROS testing requirements
  - Tooling for tests will be included in the repositories
  - Static analysis testing
  - Full DRM based scenario testing

- TSC will identify what features from ROS2 can be incorporated into Space ROS and provide prioritization
License

- Space ROS will adopt ROS2 licenses
  - Largely Apache 2.0 and Space ROS will continue to
    - Permissive
    - Commercially Useable
  - Some Creative Commands and BSD licenses for tools and documentation
Acknowledgments

This work was funded as a collaborative effort between NASA Goddard Space Flight Center, Ames Research Center, Johnson Space Center, and Blue Origin.