spaceROS

An open-source framework for developing flight-quality robotic and autonomous space systems

Ivan Perez
KBR @ NASA Ames Research Center
ivan.perezdominguez@nasa.gov

spaceROS

An open-source framework for developing flight-quality robotic and autonomous space systems

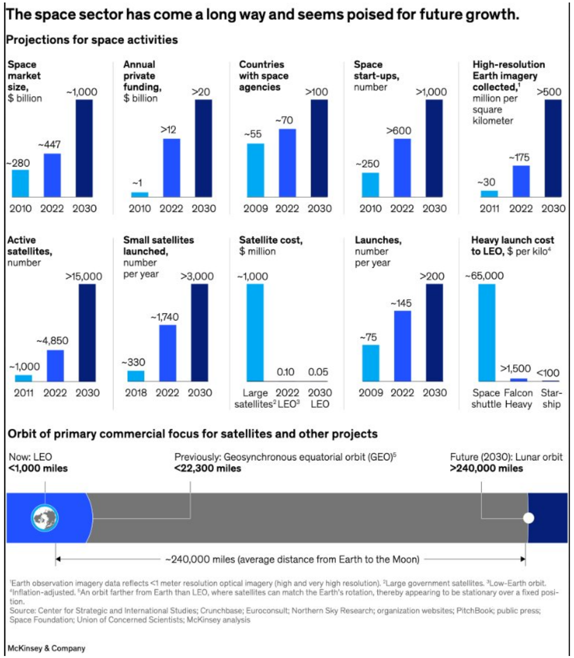
Ivan Perez
KBR @ NASA Ames Research Center
ivan.perezdominguez@nasa.gov

Agenda

- Motivation
- Goals
- What is Space ROS?
- Status
- Next steps

Space Industry

- Space sector growing rapidly
- Projected by 2030:
  - $1T market size
  - $20B in private funding
  - **> 100 countries with space agencies**
  - **> 1000 Space startups**
  - 3x number of satellites by 2030
- Robotics is Key for Long-term Space Operations



The space sector has come a long way and seems poised for future growth.

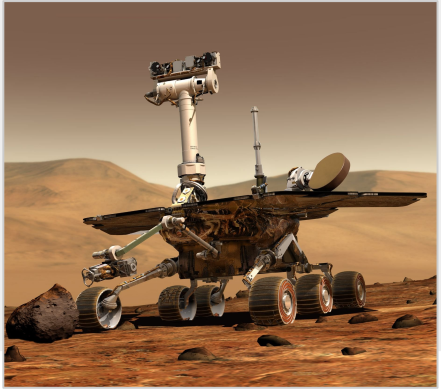
Projections for space activities

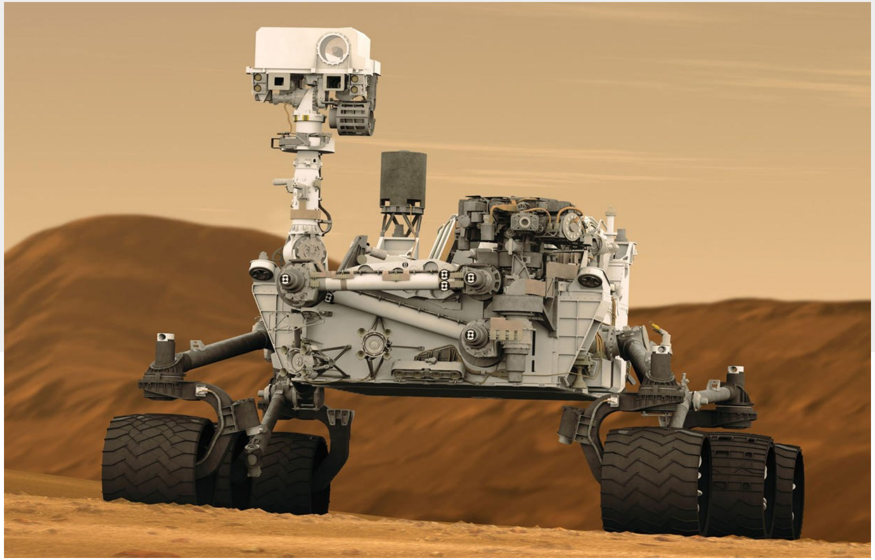
[1] Earth observation imagery data reflects <1 meter resolution optical imagery (high and very high resolution). [2] Large government satellites. [3] Low-Earth orbit. [4] Inflation-adjusted. [5] An orbit farther from Earth than LEO, where satellites can match the Earth's rotation, thereby appearing to be stationary over a fixed position.

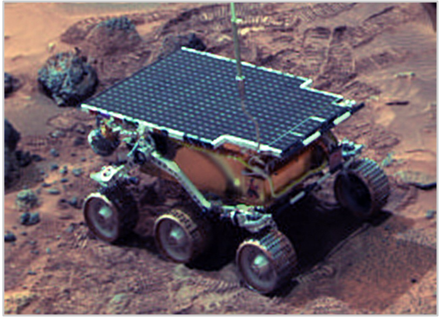
Source: Center for Strategic and International Studies; Crunchbase; Euroconsult; Northern Sky Research; organization websites; PitchBook; public press; Space Foundation; Union of Concerned Scientists; McKinsey analysis

McKinsey & Company

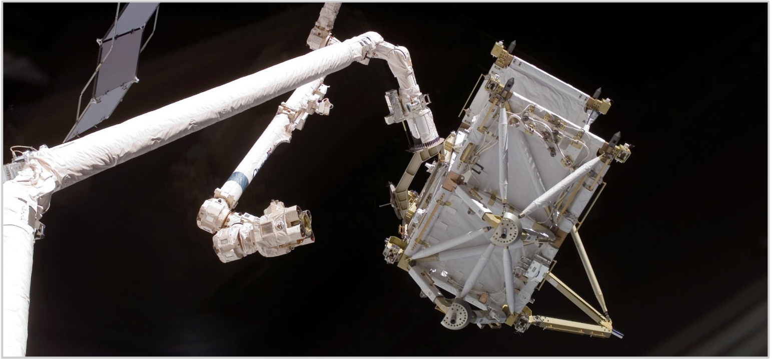
Source: https://www.mckinsey.com/featured-insights/sustainable-inclusive-growth/chart-of-the-day/a-giant-leap-for-the-space-industry

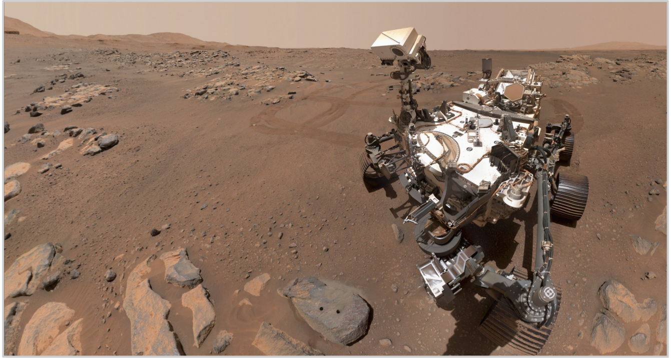


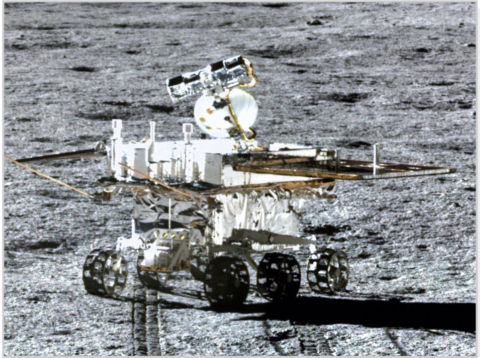




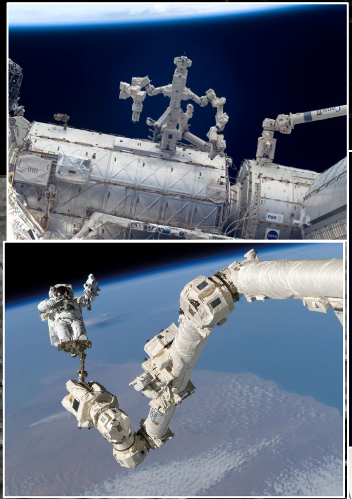
Robots are already in space



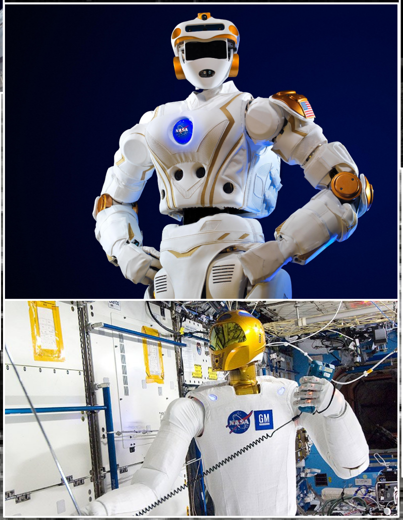




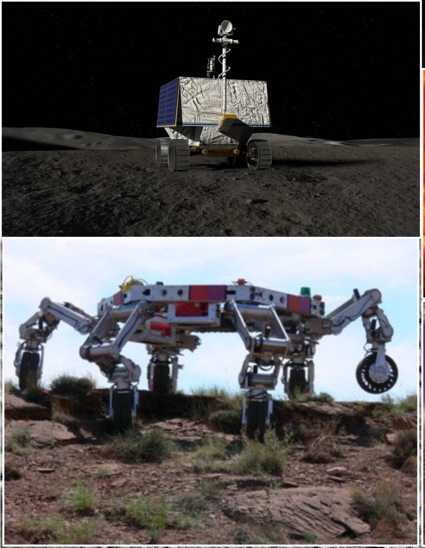
Manipulators



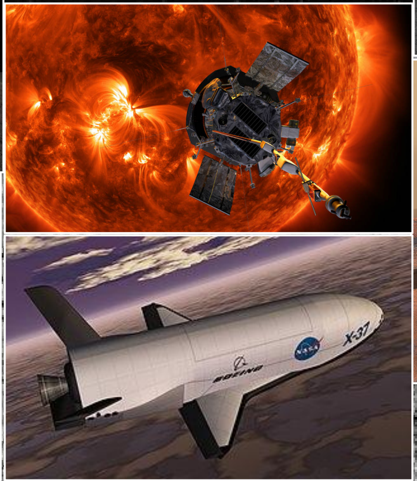
Humanoid robots



Mobility systems



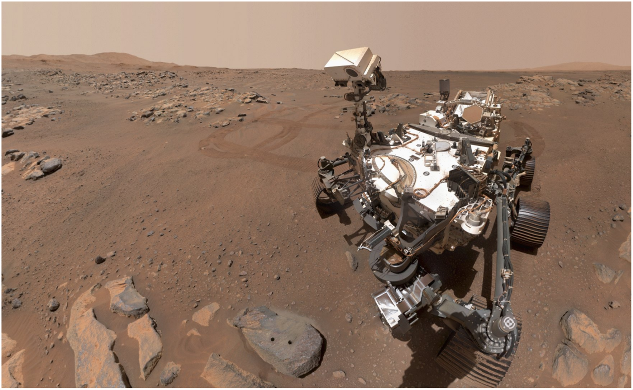
Robotic spacecraft



Robotic landers

1970 – joystick control

2020 – 2M lines of code



Increasing amount of software **+** cost of software development

= *demand for reuse*

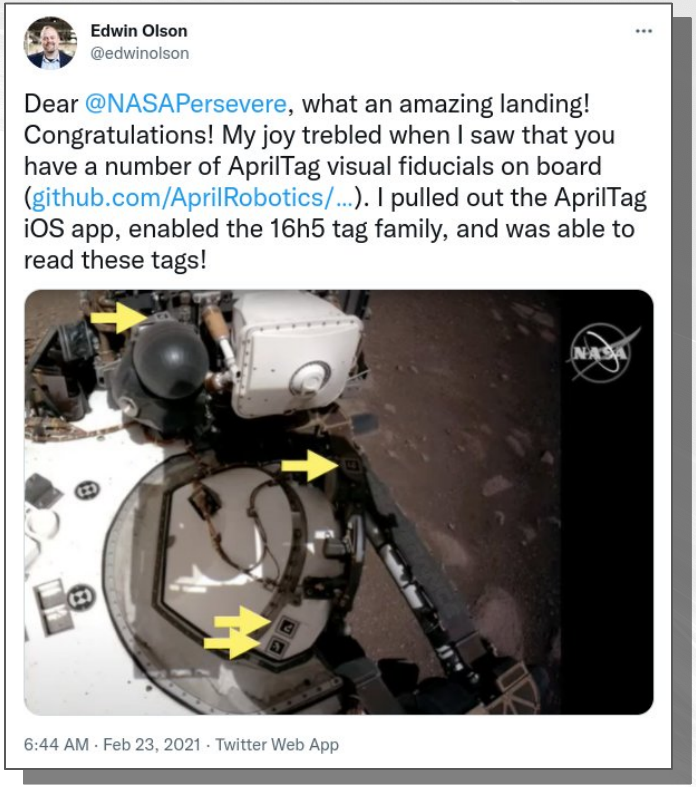
The Demand for Reuse

The space community is moving toward component-based, modular, reusable, and open frameworks for flight software and mission control

- core Flight System (cFS) – aerospace applications
- Robot Operating System -- robotics applications
- F' (F Prime) – aerospace applications
- Yamcs – command / control of spacecraft
- OpenMCT – data visualization

Also using smaller open source projects in flight

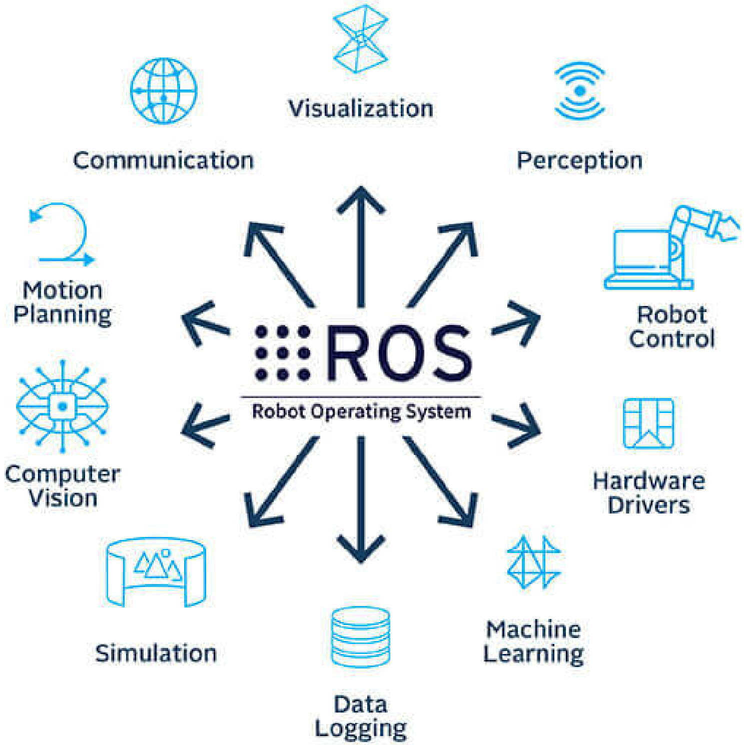
- AprilTag (visual fiducial system) used on Perseverance



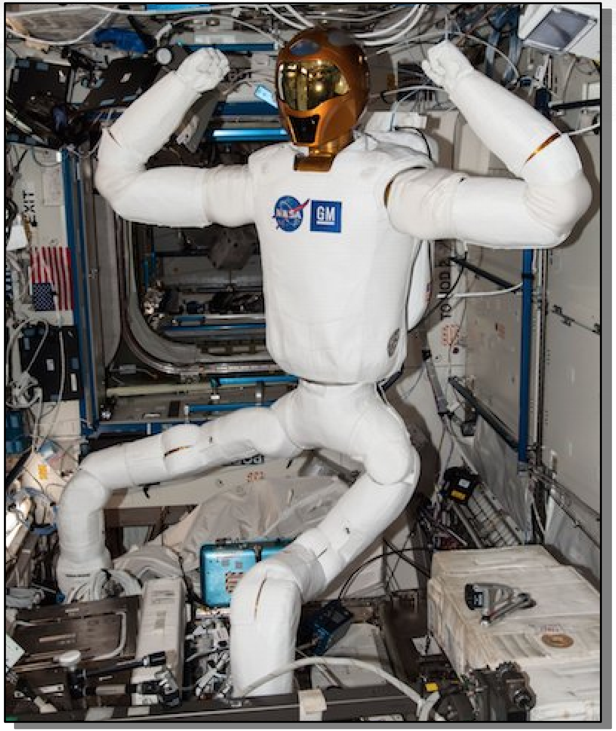
Edwin Olson
@edwinolson

Dear @NASAPersevere, what an amazing landing! Congratulations! My joy trebled when I saw that you have a number of AprilTag visual fiducials on board (github.com/AprilRobotics/...). I pulled out the AprilTag iOS app, enabled the 16h5 tag family, and was able to read these tags!

6:44 AM · Feb 23, 2021 · Twitter Web App

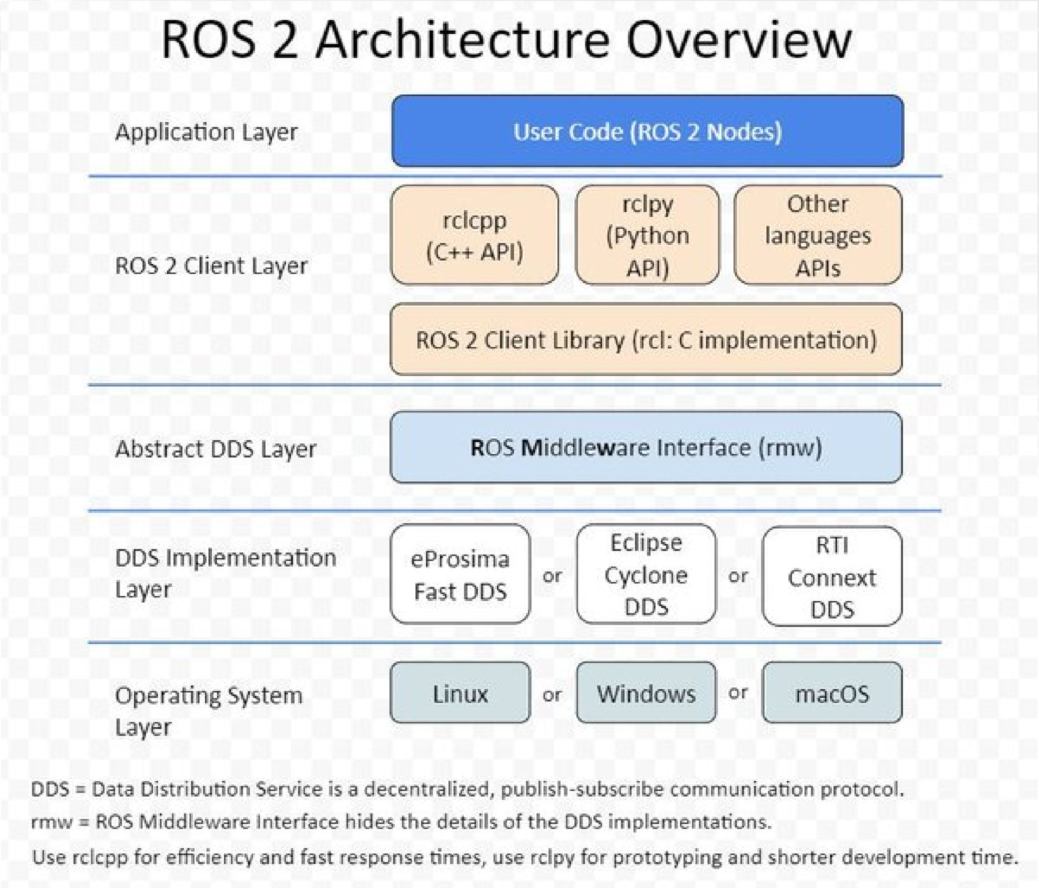

Visualization
Communication
Perception
Motion Planning
ROS
Robot Operating System
Robot Control
Computer Vision
Hardware Drivers
Simulation
Machine Learning
Data Logging

ROS-based robots have already been to space



2014: **Robonaut 2**

2019: **Astrobee**



ROS 2 Architecture Overview

Application Layer
User Code (ROS 2 Nodes)

ROS 2 Client Layer
rclcpp (C++ API)
rclpy (Python API)
Other languages APIs
ROS 2 Client Library (rcl: C implementation)

Abstract DDS Layer
ROS Middleware Interface (rmw)

DDS Implementation Layer
eProsima Fast DDS or Eclipse Cyclone DDS or RTI Connext DDS

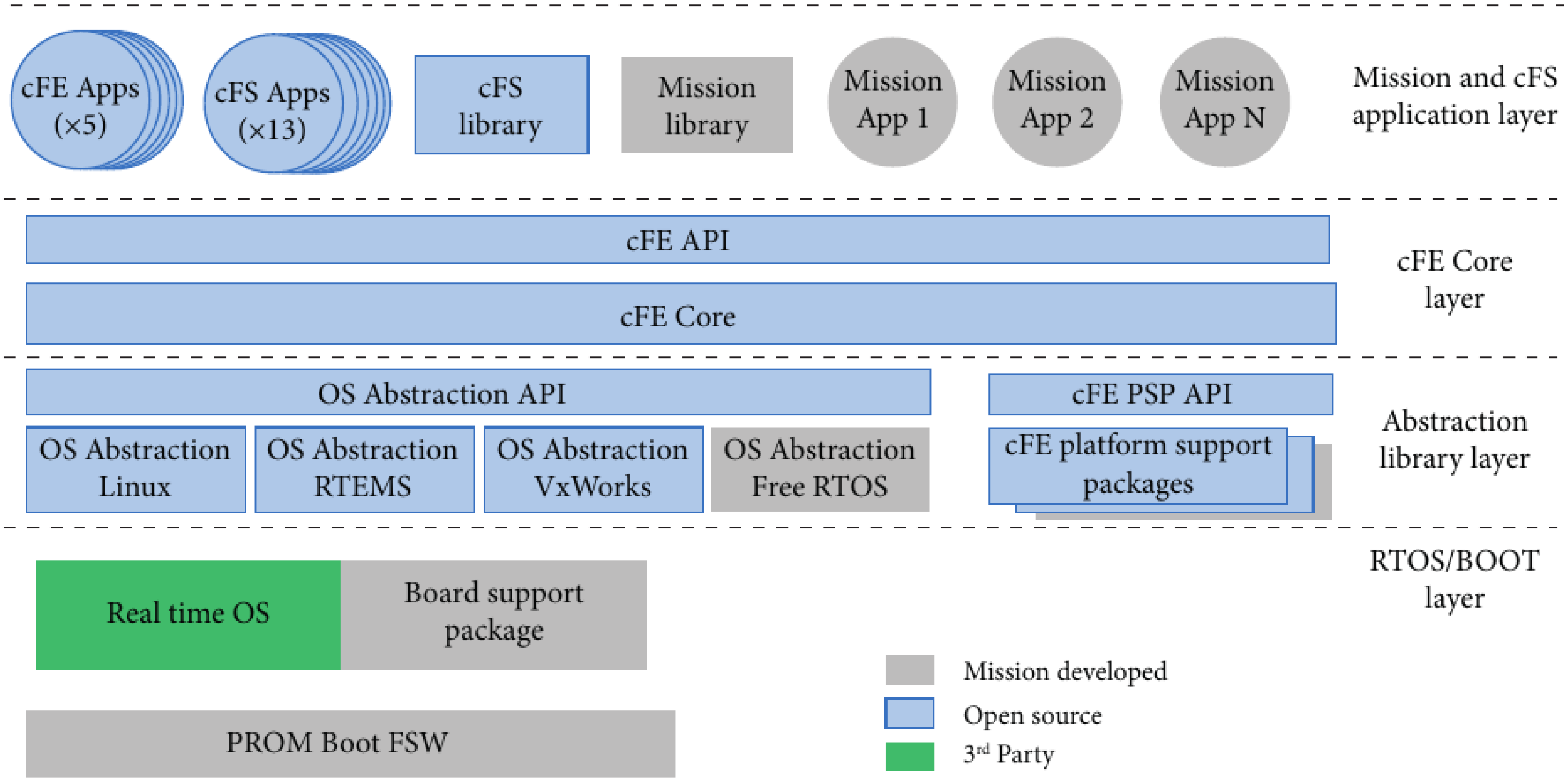
Operating System Layer
Linux or Windows or macOS

DDS = Data Distribution Service is a decentralized, publish-subscribe communication protocol.

rmw = ROS Middleware Interface hides the details of the DDS implementations.

Use rclcpp for efficiency and fast response times, use rclpy for prototyping and shorter development time.

Source: https://automaticaddison.com/ros-2-architecture-overview/



Mission and cFS application layer

cFE Apps (×5)

cFS Apps (×13)

cFS library

Mission library

Mission App 1

Mission App 2

Mission App N

cFE Core layer

cFE API

cFE Core

Abstraction library layer

OS Abstraction API

OS Abstraction Linux

OS Abstraction RTEMS

OS Abstraction VxWorks

OS Abstraction Free RTOS

cFE PSP API

cFE platform support packages

RTOS/BOOT layer

Real time OS

Board support package

PROM Boot FSW

▨ Mission developed

🟦 Open source

🟩 3rd Party

NASA cFS Software Layers and Components. Source: NASA, 2014.

Space ROS

An open-source space robotics framework for developing flight-quality robotic and autonomous space systems

spaceROS

A space-certifiable and reusable robotics framework

- Facilitate **reuse** across missions, reducing development effort and costs
- Support **certification** to flight software standards, like NASA’s NPR7150.2 and DO-178C
- Provide **artefacts** to allow space robotics projects to gain a head start
- **Aligned** with NASA so that it can be adopted for missions
- Enable **rapid development** of new robotic capabilities that reduces cost and time-to-market
- Based on **open** community, frameworks, and standards

What is Space ROS?

Procedure

- Planning Committee
- Technical Committee
- Standards Group

Technology

- Earthly/Docker images
- Sample applications
- Simulation tools & assets
- V&V tools
- Continuous Integration
- Dashboard
- Process compliance tools
- Embedded targets

Infrastructure

- Website
- Code repository
- CI server
- Docker hub
- Forums
- Documentation

Community

- Outreach
- Fixes and backports to ROS 2 project
- Community engagement

Committees

Funded in part by the Announcement of Collaboration Opportunity (ACO) program within NASA's Space Technology Mission Directorate and Blue Origin Advanced Development Programs

At the end of the ACO, OSRF designated PickNik Robotics as lead of the Planning and Technical Committees. PickNik is also using Space ROS in Phase I SBIR currently ongoing with NASA JSC.

Other members of the planning and technical committees include members from NASA, JAXA, KBR, Technology Innovation Institute (UAE), Traclabs, Emergent Space, Motiv, OSRF, Lockheed Martin, SWRI, and others.

*The mention of these companies does not imply endoresement of, or by, any of the institutuions or organizations mention.

What is Space ROS?

Procedure

- Planning Committee
- Technical Committee
- Standards Group

Technology

- Earthly/Docker images
- Sample applications
- Simulation tools & assets
- V&V tools
- Continuous Integration
- Dashboard
- Process compliance tools
- Embedded targets

Infrastructure

- Website
- Code repository
- CI server
- Docker hub
- Forums
- Documentation

Community

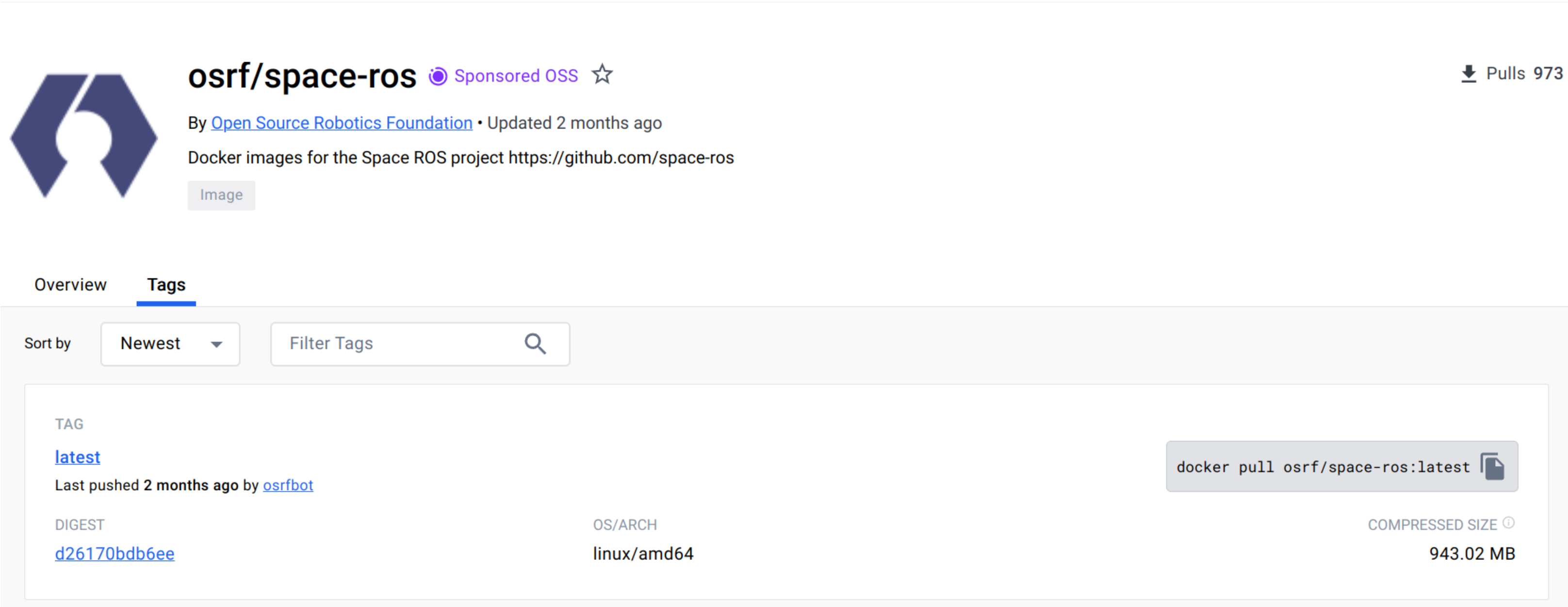
- Outreach
- Fixes and backports to ROS 2 project
- Community engagement

Technology (ongoing)

Item	Description
Docker images	Docker images provide a reproducible environment used by developers, our own CI infrastructure, and our demos.
Sample applications	Incorporate navigation (Curiosity Rover) and manipulation (Candarm) demo applications.
Simulation tools & assets	Incorporate space-related simulation assets that can then be available for use by Space ROS code.
V&V Tools	Code analysis, static analysis, assurance, etc. Some of the tools are integrated with SARIF and the build process, so that errors reported by the tools can be traced back to the code and accessed directly in VS Code.
Dashboard	Integration of SARIF output from analysis tools into VS Code.
Process compliance tools	Tools that analyze compliance with NPR 7150.2 and report potential violations.

Space ROS Docker Images

https://hub.docker.com/r/osrf/space-ros



osrf/space-ros Sponsored OSS ☆

Pulls 973

By Open Source Robotics Foundation • Updated 2 months ago

Docker images for the Space ROS project https://github.com/space-ros

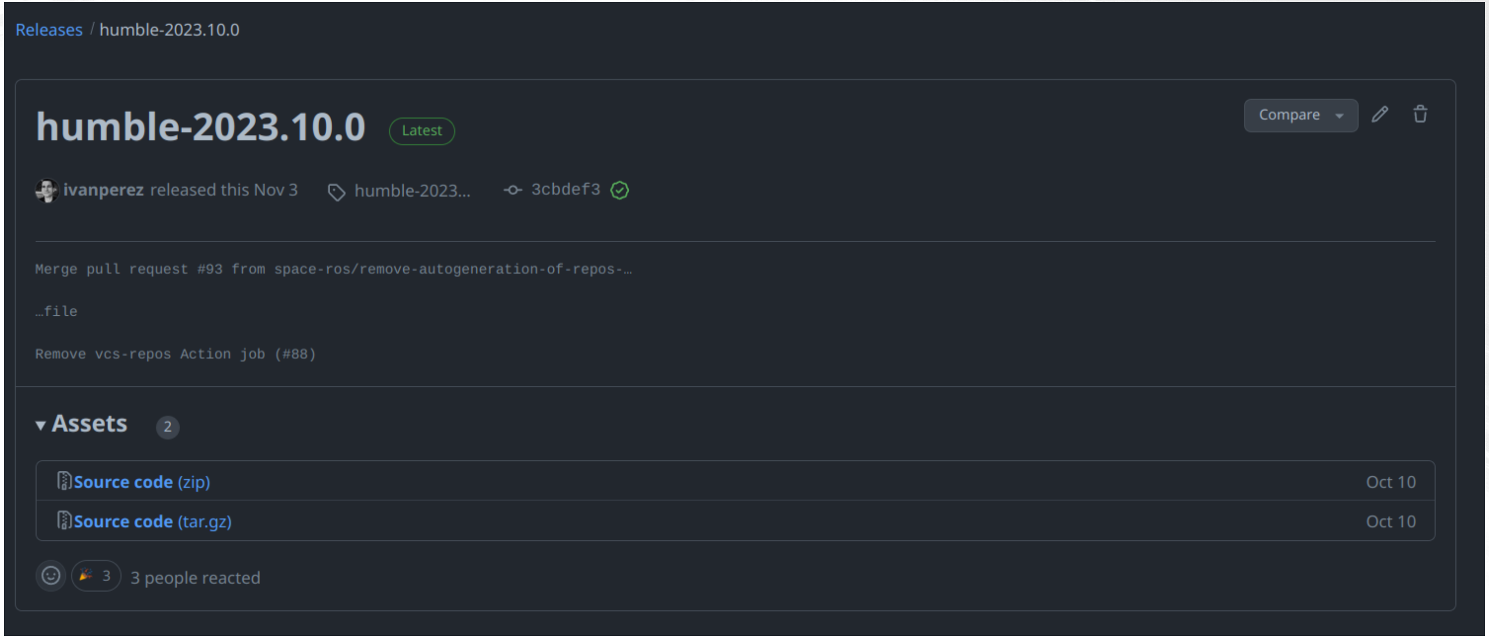
Image

Overview Tags

Sort by Newest ▾ Filter Tags

TAG		
latest Last pushed **2 months ago** by osrfbot		docker pull osrf/space-ros:latest
DIGEST	OS/ARCH	COMPRESSED SIZE
d26170bdb6ee	linux/amd64	943.02 MB

Space ROS Releases



Releases / humble-2023.10.0

humble-2023.10.0 Latest

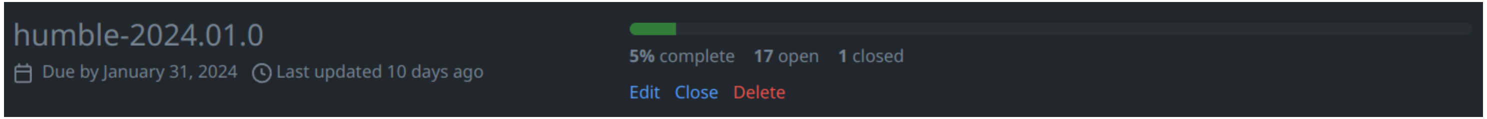
ivanperez released this Nov 3 humble-2023... 3cbdef3

```
Merge pull request #93 from space-ros/remove-autogeneration-of-repos-…

…file

Remove vcs-repos Action job (#88)
```

Assets 2

Source code (zip)	Oct 10
Source code (tar.gz)	Oct 10

🎉 3 3 people reacted



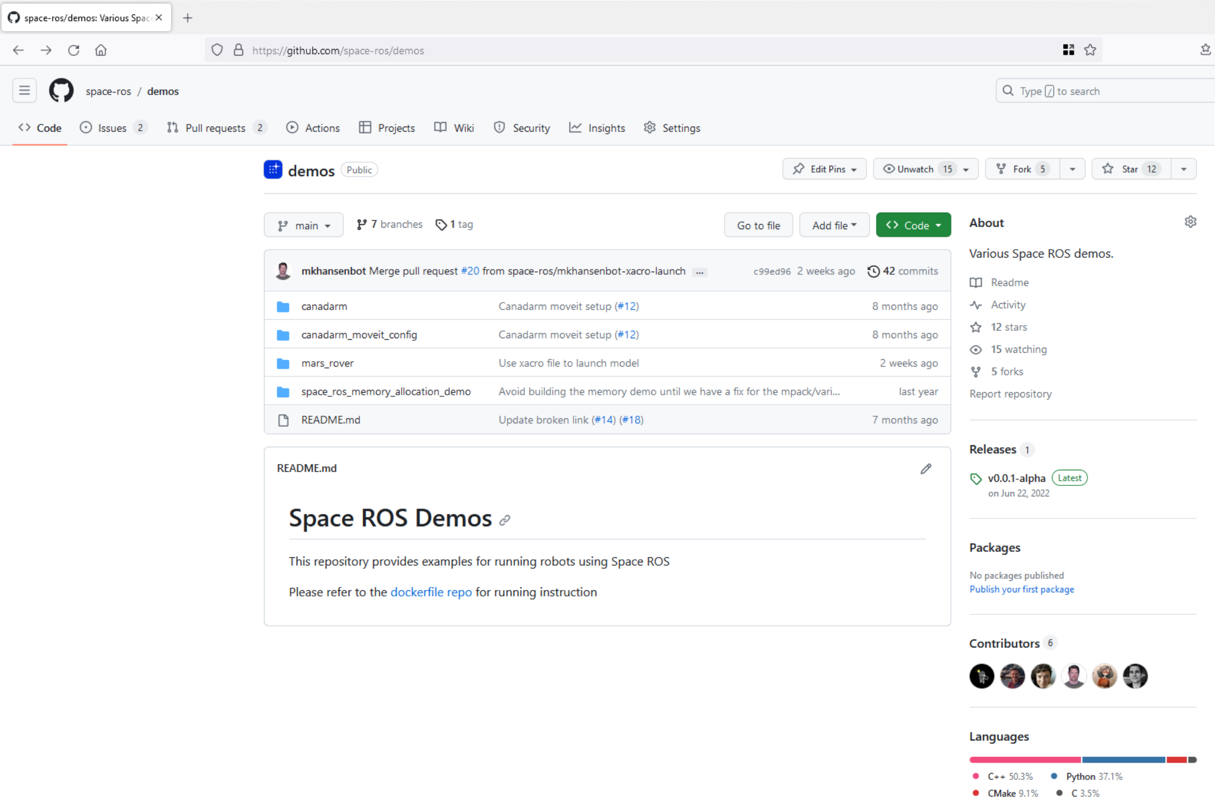
humble-2024.01.0

Due by January 31, 2024 Last updated 10 days ago

5% complete **17** open **1** closed

Space ROS Demos

https://github.com/space-ros/demos



space-ros/demos: Various Spac ×

https://github.com/space-ros/demos

space-ros / demos

Type / to search

Code | Issues 2 | Pull requests 2 | Actions | Projects | Wiki | Security | Insights | Settings

demos Public

Edit Pins | Unwatch 15 | Fork 5 | Star 12

main | 7 branches | 1 tag | Go to file | Add file | Code

mkhansenbot Merge pull request #20 from space-ros/mkhansenbot-xacro-launch … c99ed96 2 weeks ago 42 commits

canadarm	Canadarm moveit setup (#12)	8 months ago
canadarm_moveit_config	Canadarm moveit setup (#12)	8 months ago
mars_rover	Use xacro file to launch model	2 weeks ago
space_ros_memory_allocation_demo	Avoid building the memory demo until we have a fix for the mpack/vari…	last year
README.md	Update broken link (#14) (#18)	7 months ago

README.md

Space ROS Demos

This repository provides examples for running robots using Space ROS

Please refer to the dockerfile repo for running instruction

About

Various Space ROS demos.

Readme
Activity
12 stars
15 watching
5 forks
Report repository

Releases 1

v0.0.1-alpha Latest
on Jun 22, 2022

Packages

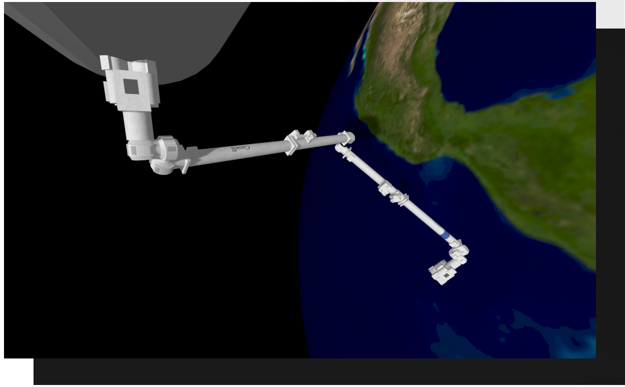
No packages published
Publish your first package

Contributors 6

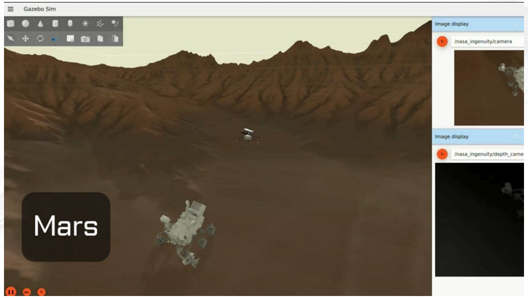
Languages

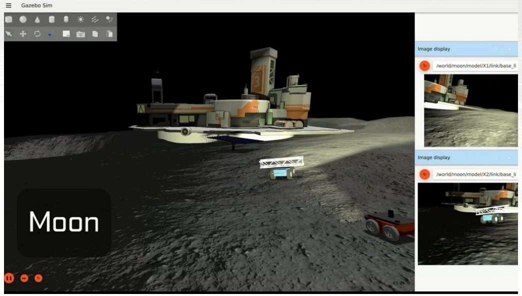
● C++ 50.3% ● Python 37.1%
● CMake 9.1% ● C 3.5%

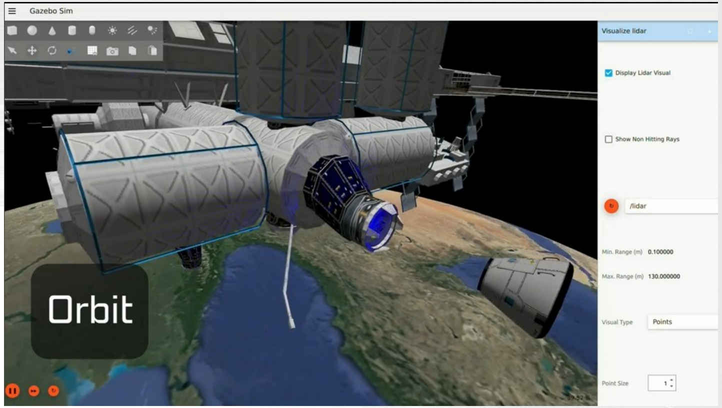
Image dis...
/image_raw



NASA Spark




Gazebo Sim
Image display
/world/moon/model/X1/link/base_li
Image display
/world/moon/model/X2/link/base_li
Moon



☑ Display Lidar Visual

□ Show Non Hitting Rays

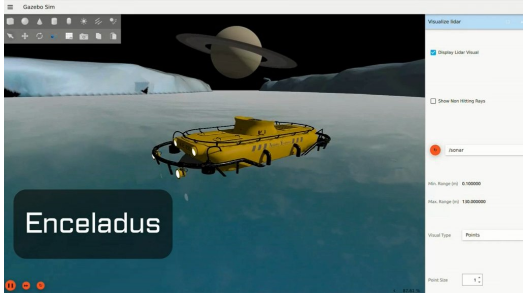
/lidar

Min. Range (m) 0.100000

Max. Range (m) 130.000000

Visual Type Points

Point Size 1



☑ Display Lidar Visual

□ Show Non Hitting Rays

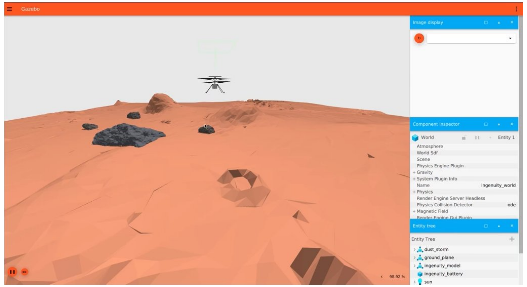
/sonar

Min. Range (m) 0.100000

Max. Range (m) 130.000000

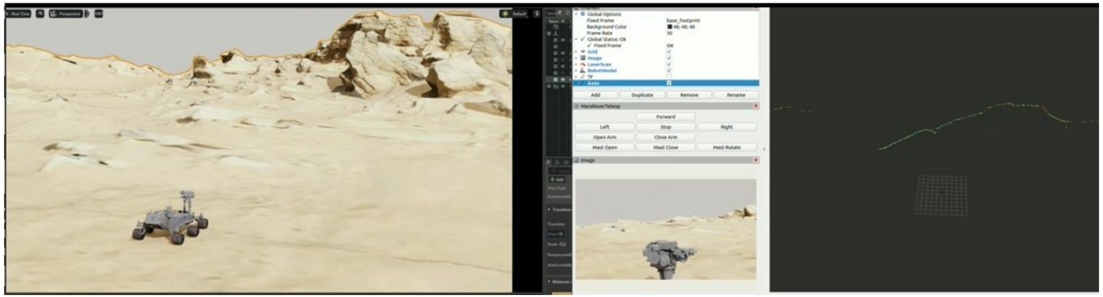
Visual Type Points

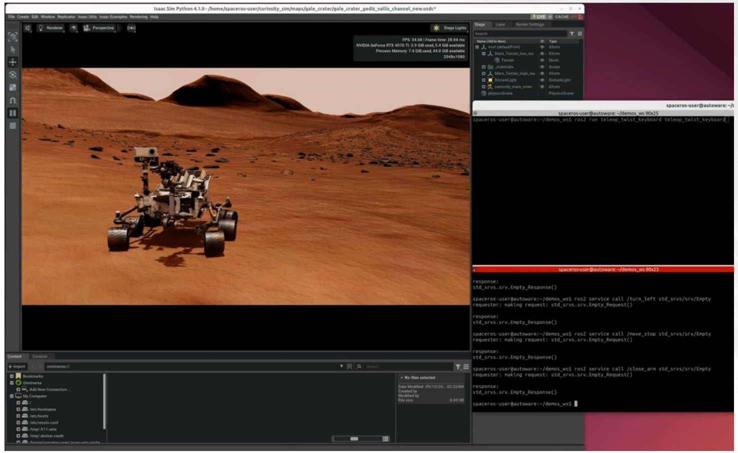
Point Size 1



Name ingenuity_world

Entity Tree
- dust_storm
- ground_plane
- ingenuity_model
- ingenuity_battery
- sun



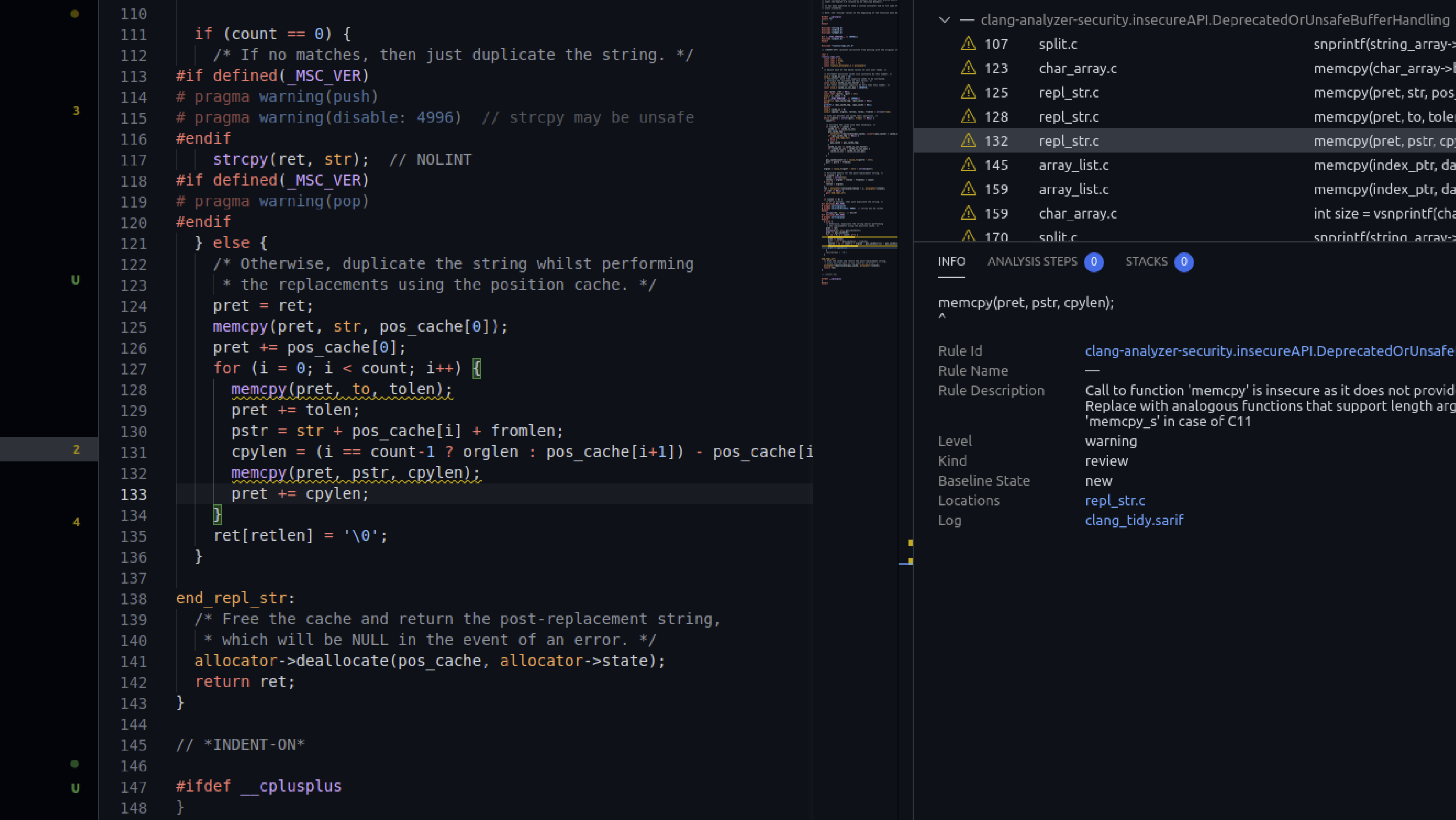

Isaac Sim Python 4.1.0 - /home/spaceros-user/curiosity_sim/maps/gale_crater/gale_crater_gediz_vallis_channel_new.usdc*

```
spaceros-user@autoware:~/demos_ws$ ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

```
response:
std_srvs.srv.Empty_Response()

spaceros-user@autoware:~/demos_ws$ ros2 service call /turn_left std_srvs/srv/Empty
requester: making request: std_srvs.srv.Empty_Request()

response:
std_srvs.srv.Empty_Response()

spaceros-user@autoware:~/demos_ws$ ros2 service call /move_stop std_srvs/srv/Empty
requester: making request: std_srvs.srv.Empty_Request()

response:
std_srvs.srv.Empty_Response()

spaceros-user@autoware:~/demos_ws$ ros2 service call /close_arm std_srvs/srv/Empty
requester: making request: std_srvs.srv.Empty_Request()

response:
std_srvs.srv.Empty_Response()

spaceros-user@autoware:~/demos_ws$ 
```

Static Analysis

- Increase code quality, ease verification
- Space ROS provides a suite of static analyzers, including IKOS and Cobra from NASA
- Currently adding dynamic analysis: code coverage and MC/DC testing
- The static analysis tools generate SARIF output
  - Most by parsing output of the tool
  - Tools should eventually support SARIF directly; would allow for more detailed information in SARIF, such as logical location
- Filtering pass to remove (some) redundancy
  - Currently, removing identical issues
  - Would like to remove semantic equivalents
- The results are made available to the Space ROS Dashboard
  - An archive format that contains analyzer output, filtered output, and metadata



```
110
111   if (count == 0) {
112     /* If no matches, then just duplicate the string. */
113 #if defined(_MSC_VER)
114 # pragma warning(push)
115 # pragma warning(disable: 4996)  // strcpy may be unsafe
116 #endif
117     strcpy(ret, str);  // NOLINT
118 #if defined(_MSC_VER)
119 # pragma warning(pop)
120 #endif
121   } else {
122     /* Otherwise, duplicate the string whilst performing
123      * the replacements using the position cache. */
124     pret = ret;
125     memcpy(pret, str, pos_cache[0]);
126     pret += pos_cache[0];
127     for (i = 0; i < count; i++) {
128       memcpy(pret, to, tolen);
129       pret += tolen;
130       pstr = str + pos_cache[i] + fromlen;
131       cpylen = (i == count-1 ? orglen : pos_cache[i+1]) - pos_cache[i
132       memcpy(pret, pstr, cpylen);
133       pret += cpylen;
134     }
135     ret[retlen] = '\0';
136   }
137
138 end_repl_str:
139   /* Free the cache and return the post-replacement string,
140    * which will be NULL in the event of an error. */
141   allocator->deallocate(pos_cache, allocator->state);
142   return ret;
143 }
144
145 // *INDENT-ON*
146
147 #ifdef __cplusplus
148 }
```

— clang-analyzer-security.insecureAPI.DeprecatedOrUnsafeBufferHandling

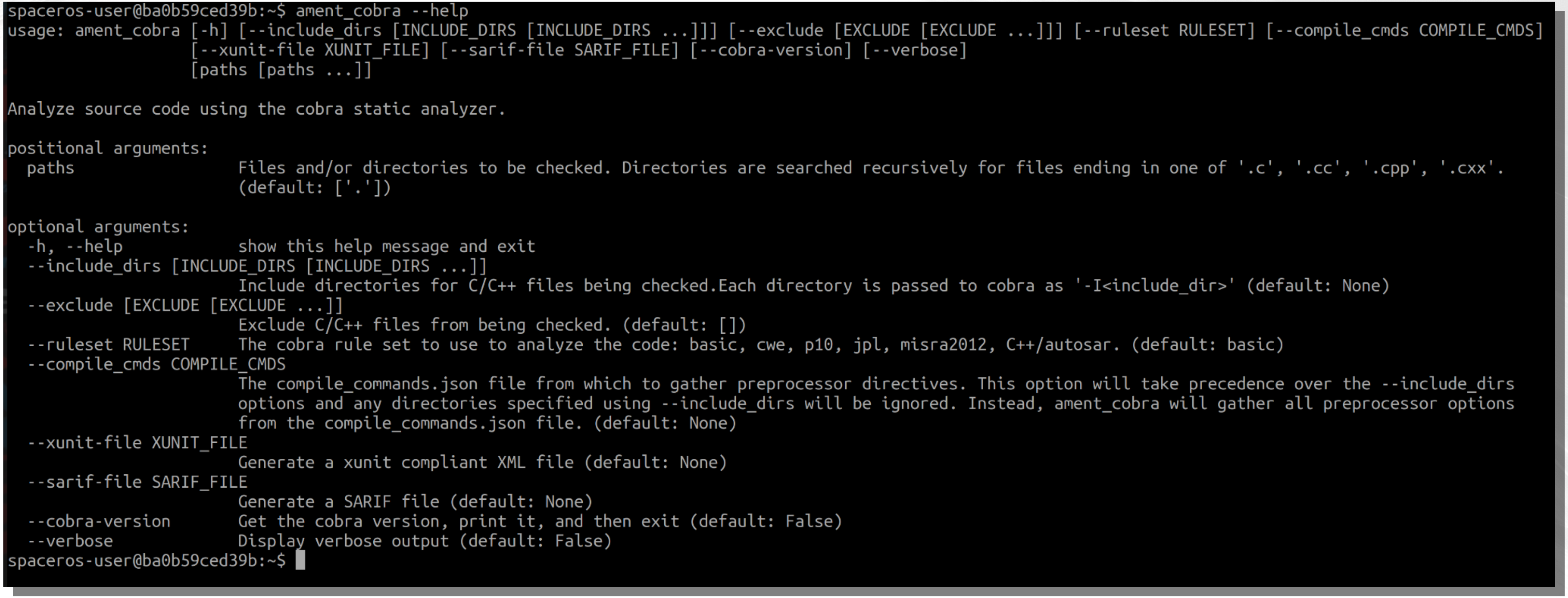
⚠ 107	split.c	snprintf(string_array->
⚠ 123	char_array.c	memcpy(char_array->b
⚠ 125	repl_str.c	memcpy(pret, str, pos_
⚠ 128	repl_str.c	memcpy(pret, to, tolen
⚠ 132	repl_str.c	memcpy(pret, pstr, cp
⚠ 145	array_list.c	memcpy(index_ptr, da
⚠ 159	array_list.c	memcpy(index_ptr, da
⚠ 159	char_array.c	int size = vsnprintf(cha
⚠ 170	split.c	snprintf(string_array->

```
memcpy(pret, pstr, cpylen);
^
```

Rule Id	clang-analyzer-security.insecureAPI.DeprecatedOrUnsafe
Rule Name	—
Rule Description	Call to function 'memcpy' is insecure as it does not provid Replace with analogous functions that support length arg 'memcpy_s' in case of C11
Level	warning
Kind	review
Baseline State	new
Locations	repl_str.c
Log	clang_tidy.sarif

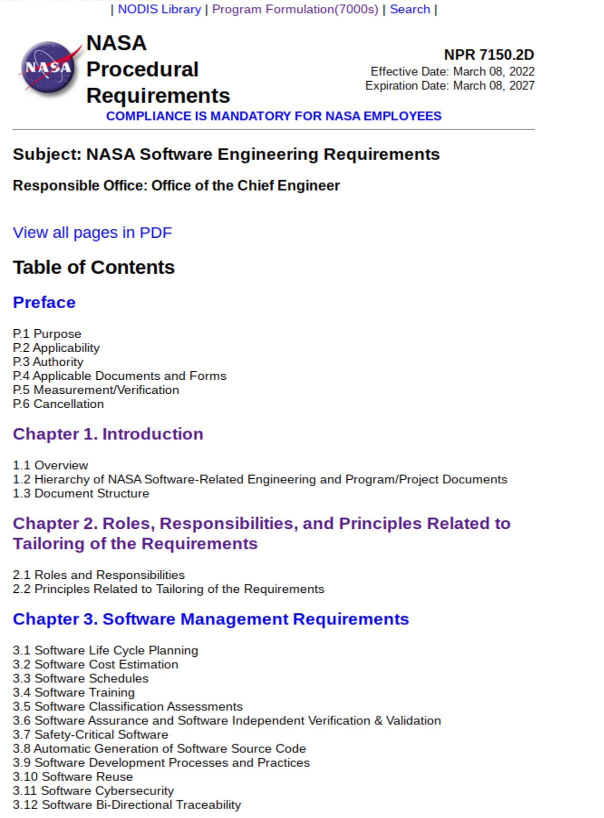
Cobra (code browser and analyzer)

An extensible, interactive tool for the analysis of C/C++ code



```
spaceros-user@ba0b59ced39b:~$ ament_cobra --help
usage: ament_cobra [-h] [--include_dirs [INCLUDE_DIRS [INCLUDE_DIRS ...]]] [--exclude [EXCLUDE [EXCLUDE ...]]] [--ruleset RULESET] [--compile_cmds COMPILE_CMDS]
                   [--xunit-file XUNIT_FILE] [--sarif-file SARIF_FILE] [--cobra-version] [--verbose]
                   [paths [paths ...]]

Analyze source code using the cobra static analyzer.

positional arguments:
  paths                 Files and/or directories to be checked. Directories are searched recursively for files ending in one of '.c', '.cc', '.cpp', '.cxx'.
                        (default: ['.'])

optional arguments:
  -h, --help            show this help message and exit
  --include_dirs [INCLUDE_DIRS [INCLUDE_DIRS ...]]
                        Include directories for C/C++ files being checked.Each directory is passed to cobra as '-I<include_dir>' (default: None)
  --exclude [EXCLUDE [EXCLUDE ...]]
                        Exclude C/C++ files from being checked. (default: [])
  --ruleset RULESET     The cobra rule set to use to analyze the code: basic, cwe, p10, jpl, misra2012, C++/autosar. (default: basic)
  --compile_cmds COMPILE_CMDS
                        The compile_commands.json file from which to gather preprocessor directives. This option will take precedence over the --include_dirs
                        options and any directories specified using --include_dirs will be ignored. Instead, ament_cobra will gather all preprocessor options
                        from the compile_commands.json file. (default: None)
  --xunit-file XUNIT_FILE
                        Generate a xunit compliant XML file (default: None)
  --sarif-file SARIF_FILE
                        Generate a SARIF file (default: None)
  --cobra-version       Get the cobra version, print it, and then exit (default: False)
  --verbose             Display verbose output (default: False)
spaceros-user@ba0b59ced39b:~$ 
```

Process compliance



| NODIS Library | Program Formulation(7000s) | Search |

NASA
Procedural
Requirements

NPR 7150.2D
Effective Date: March 08, 2022
Expiration Date: March 08, 2027

COMPLIANCE IS MANDATORY FOR NASA EMPLOYEES

Subject: NASA Software Engineering Requirements

Responsible Office: Office of the Chief Engineer

View all pages in PDF

Table of Contents

Preface

P.1 Purpose
P.2 Applicability
P.3 Authority
P.4 Applicable Documents and Forms
P.5 Measurement/Verification
P.6 Cancellation

Chapter 1. Introduction

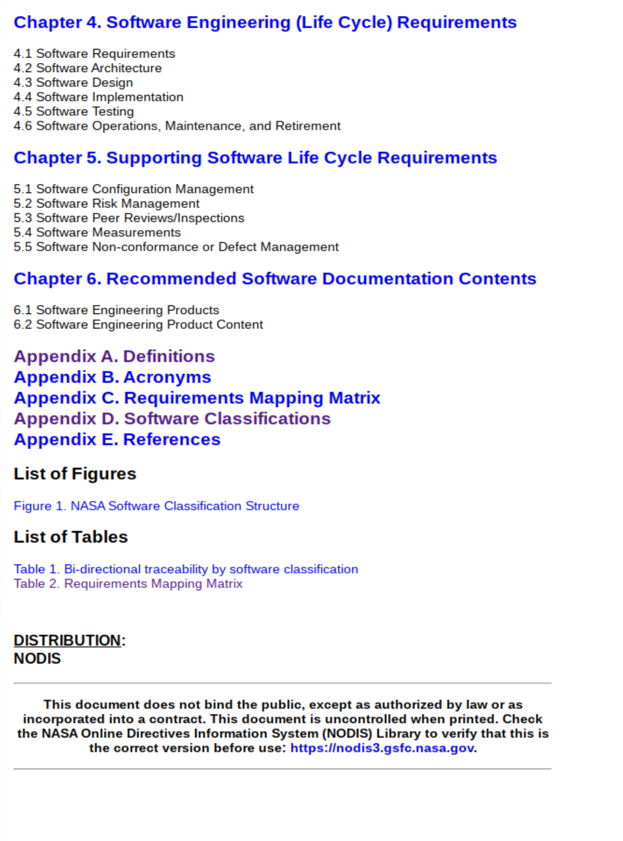
1.1 Overview
1.2 Hierarchy of NASA Software-Related Engineering and Program/Project Documents
1.3 Document Structure

Chapter 2. Roles, Responsibilities, and Principles Related to Tailoring of the Requirements

2.1 Roles and Responsibilities
2.2 Principles Related to Tailoring of the Requirements

Chapter 3. Software Management Requirements

3.1 Software Life Cycle Planning
3.2 Software Cost Estimation
3.3 Software Schedules
3.4 Software Training
3.5 Software Classification Assessments
3.6 Software Assurance and Software Independent Verification & Validation
3.7 Safety-Critical Software
3.8 Automatic Generation of Software Source Code
3.9 Software Development Processes and Practices
3.10 Software Reuse
3.11 Software Cybersecurity
3.12 Software Bi-Directional Traceability



Chapter 4. Software Engineering (Life Cycle) Requirements

4.1 Software Requirements
4.2 Software Architecture
4.3 Software Design
4.4 Software Implementation
4.5 Software Testing
4.6 Software Operations, Maintenance, and Retirement

Chapter 5. Supporting Software Life Cycle Requirements

5.1 Software Configuration Management
5.2 Software Risk Management
5.3 Software Peer Reviews/Inspections
5.4 Software Measurements
5.5 Software Non-conformance or Defect Management

Chapter 6. Recommended Software Documentation Contents

6.1 Software Engineering Products
6.2 Software Engineering Product Content

Appendix A. Definitions
Appendix B. Acronyms
Appendix C. Requirements Mapping Matrix
Appendix D. Software Classifications
Appendix E. References

List of Figures

Figure 1. NASA Software Classification Structure

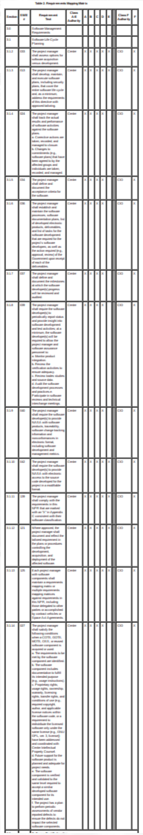
List of Tables

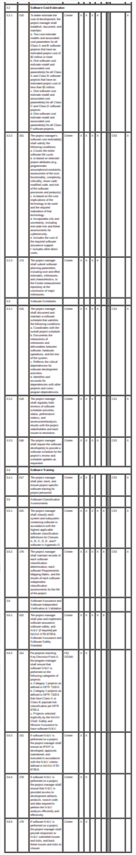
Table 1. Bi-directional traceability by software classification
Table 2. Requirements Mapping Matrix

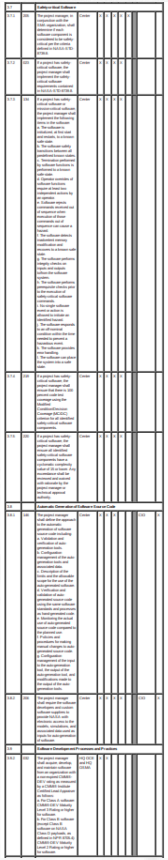
DISTRIBUTION:
NODIS

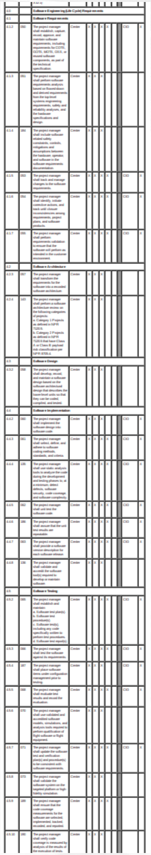
This document does not bind the public, except as authorized by law or as incorporated into a contract. This document is uncontrolled when printed. Check the NASA Online Directives Information System (NODIS) Library to verify that this is the correct version before use: https://nodis3.gsfc.nasa.gov.

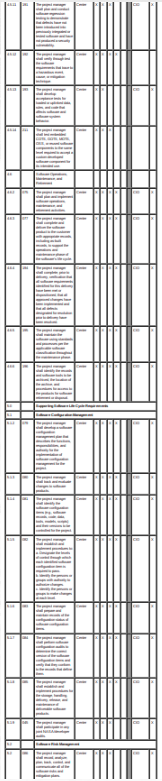
NPR 7150

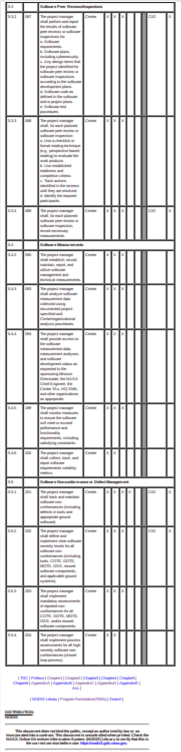




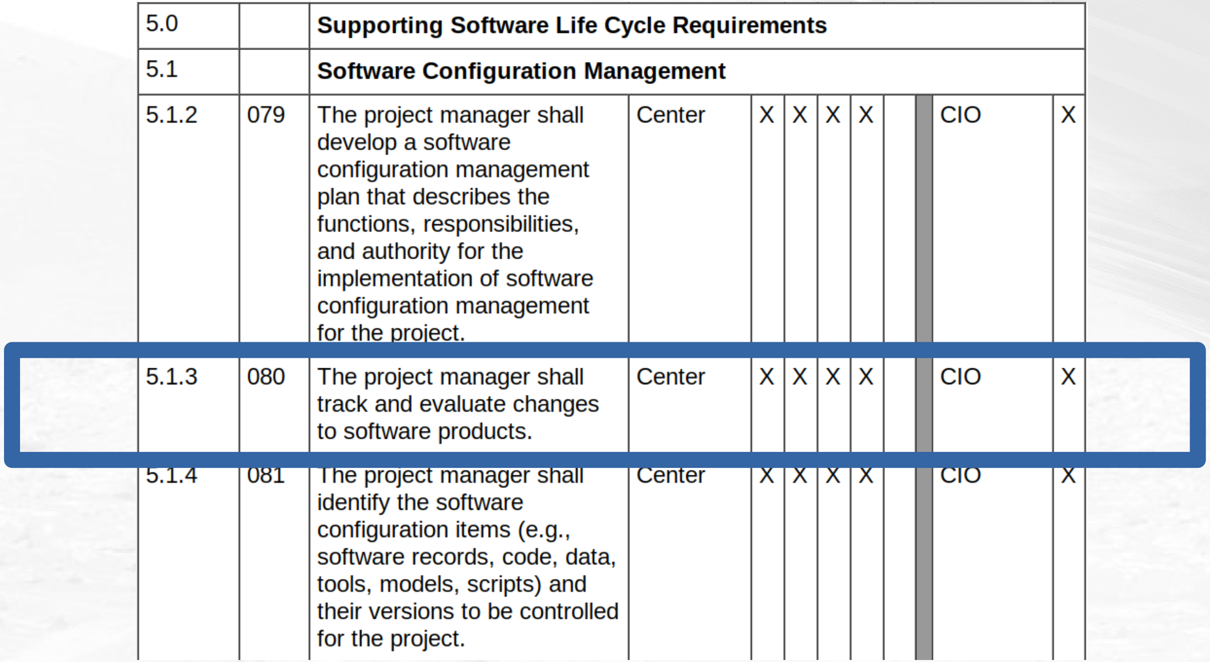






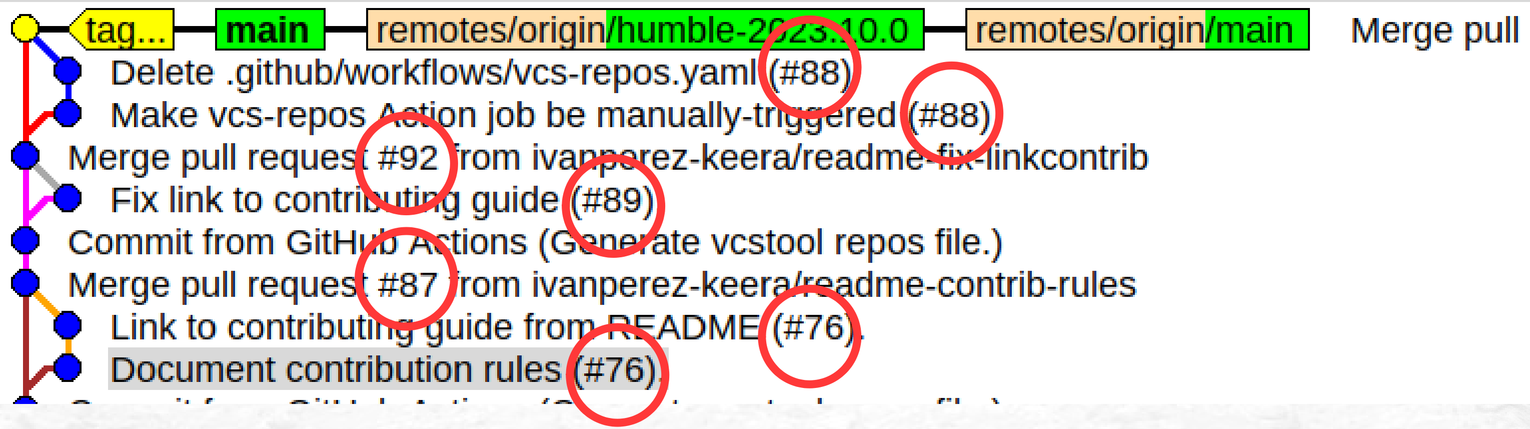


NPR 7150



5.0		**Supporting Software Life Cycle Requirements**									
5.1		**Software Configuration Management**									
5.1.2	079	The project manager shall develop a software configuration management plan that describes the functions, responsibilities, and authority for the implementation of software configuration management for the project.	Center	X	X	X	X		▨	CIO	X
5.1.3	080	The project manager shall track and evaluate changes to software products.	Center	X	X	X	X		▨	CIO	X
5.1.4	081	The project manager shall identify the software configuration items (e.g., software records, code, data, tools, models, scripts) and their versions to be controlled for the project.	Center	X	X	X	X		▨	CIO	X

Traceability: CRs to Code



```
tag...  main  remotes/origin/humble-2023.10.0  remotes/origin/main  Merge pull
    Delete .github/workflows/vcs-repos.yaml (#88)
    Make vcs-repos Action job be manually-triggered (#88)
Merge pull request #92 from ivanperez-keera/readme-fix-linkcontrib
    Fix link to contributing guide (#89)
Commit from GitHub Actions (Generate vcstool repos file.)
Merge pull request #87 from ivanperez-keera/readme-contrib-rules
    Link to contributing guide from README (#76).
    Document contribution rules (#76).
```

Process Compliance

- Process (DONE)
- Auditing via scripts, running in CI (started)
- Auditing using reporting tools (ongoing, more later) (currently NASA only)

Current Status (Dec 2024)

- 5 releases published at regular intervals (quarterly releases)
- Next release planned for Jan 2025

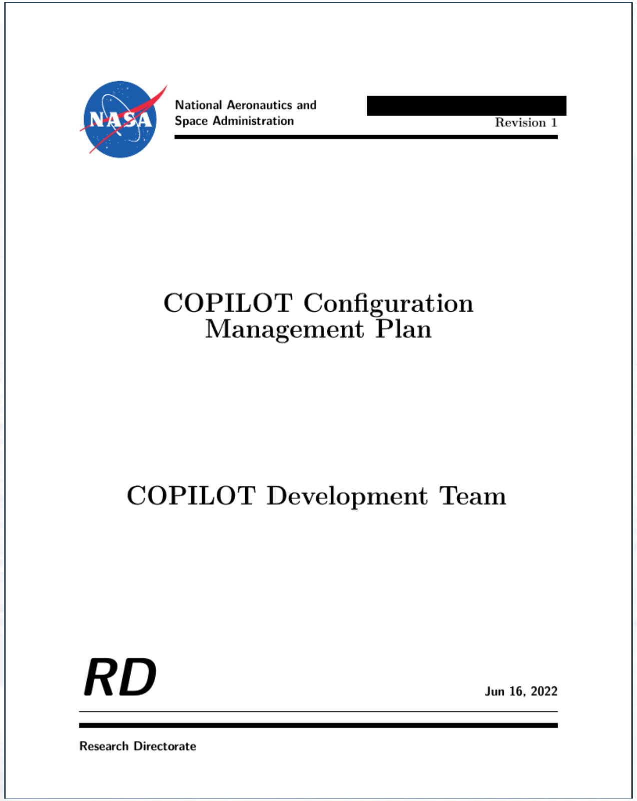
Current focus (Dec 2024)

- Simplifying the development process.
- Support for other architectures.
- Documenting development, maintenance.
- Adding better integration of static analysis and V&V tools.
- Evaluating and facilitating compliance with NPR7150.2.
- Integrating demos from NASA Spark challenge.
- Increasing participation from the community.
- Working on integration with cFS and FPrime
  - Traclabs bridge.
  - JAXA bridge.

NASA/TM–XXXXXX

NPR7150.2 Compliance in Space ROS

Documentation



National Aeronautics and
Space Administration

[redacted]

Revision 1

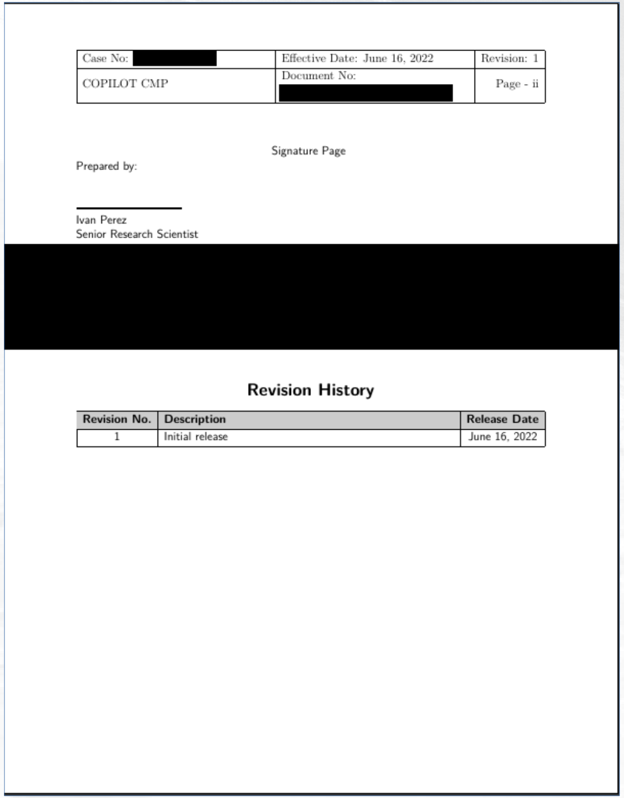
COPILOT Configuration Management Plan

COPILOT Development Team

RD

Jun 16, 2022

Research Directorate



Case No: [redacted]	Effective Date: June 16, 2022	Revision: 1
COPILOT CMP	Document No: [redacted]	Page - ii

Signature Page

Prepared by:

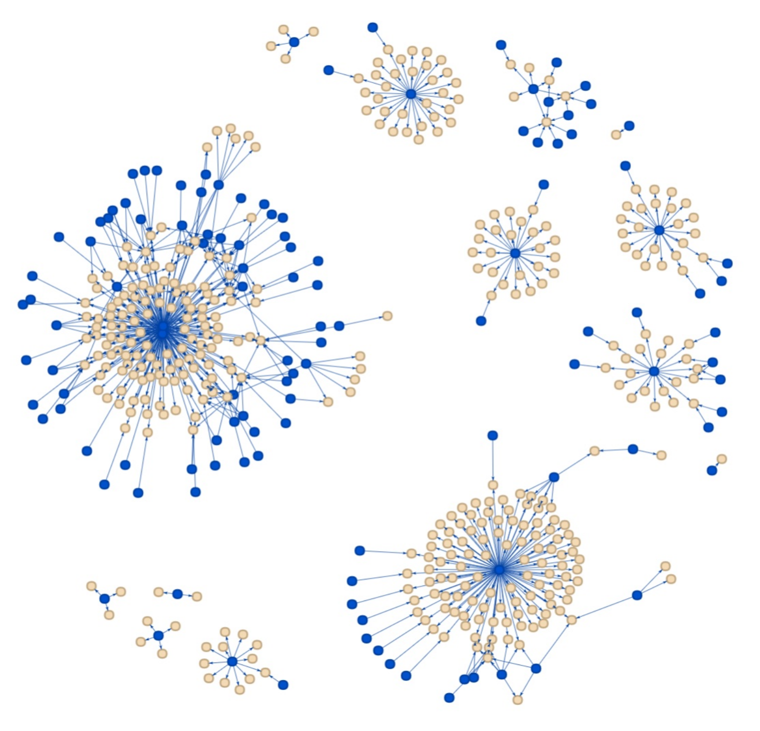
Ivan Perez
Senior Research Scientist

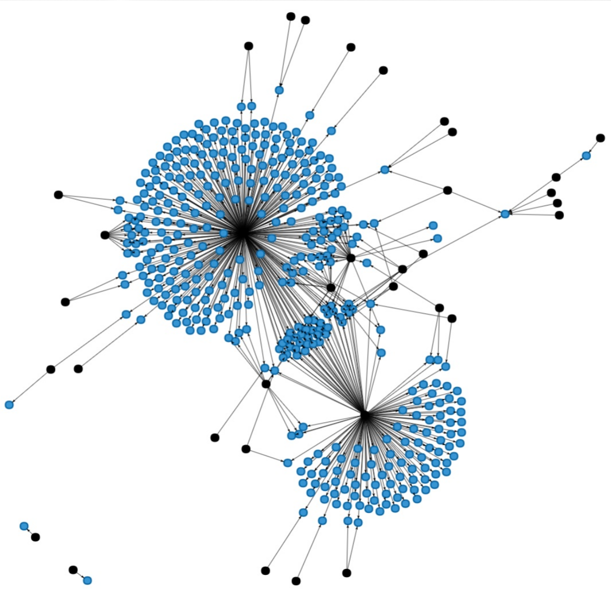
[redacted]

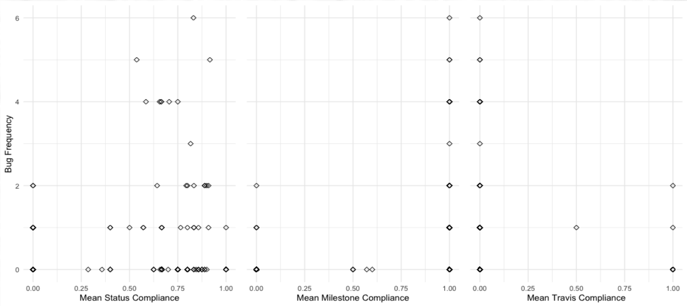
Revision History

Revision No.	Description	Release Date
1	Initial release	June 16, 2022

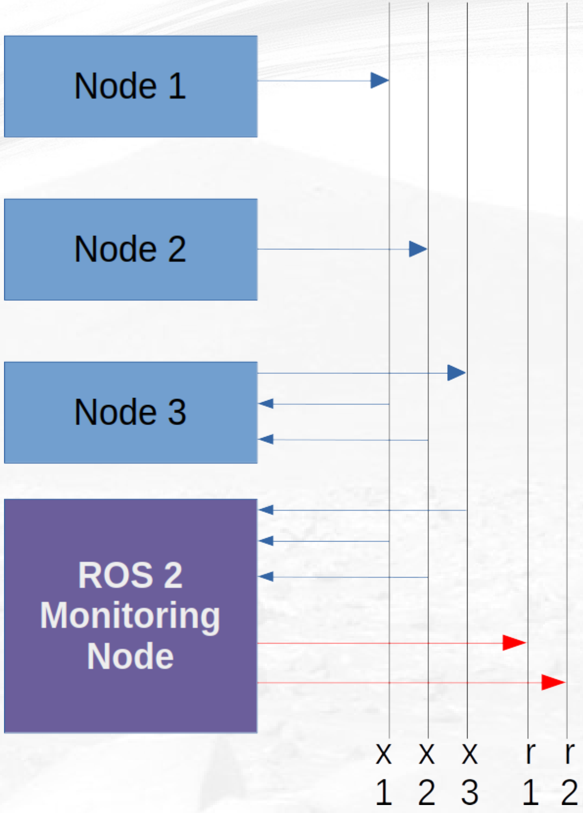
Automated Auditing

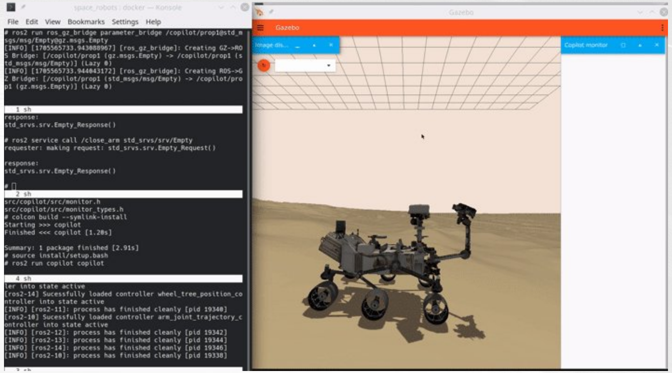






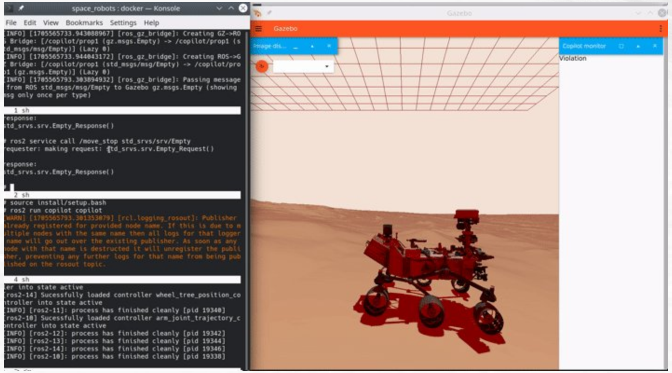
Runtime Monitoring of Space ROS with Copilot





space_robots : docker — Konsole
File Edit View Bookmarks Settings Help
Gazebo
Image dis...
Copilot monitor

```
# ros2 run ros_gz_bridge parameter_bridge /copilot/prop1@std_m
sgs/msg/Empty@gz.msgs.Empty
[INFO] [1705565733.943088967] [ros_gz_bridge]: Creating GZ->RO
S Bridge: [/copilot/prop1 (gz.msgs.Empty) -> /copilot/prop1 (s
td_msgs/msg/Empty)] (Lazy 0)
[INFO] [1705565733.944043172] [ros_gz_bridge]: Creating ROS->G
Z Bridge: [/copilot/prop1 (std_msgs/msg/Empty) -> /copilot/pro
p1 (gz.msgs.Empty)] (Lazy 0)

   1 sh
response:
std_srvs.srv.Empty_Response()

# ros2 service call /close_arm std_srvs/srv/Empty
requester: making request: std_srvs.srv.Empty_Request()

response:
std_srvs.srv.Empty_Response()

# 
   2 sh
src/copilot/src/monitor.h
src/copilot/src/monitor_types.h
# colcon build --symlink-install
Starting >>> copilot
Finished <<< copilot [1.20s]

Summary: 1 package finished [2.91s]
# source install/setup.bash
# ros2 run copilot copilot

   4 sh
ler into state active
[ros2-14] Sucessfully loaded controller wheel_tree_position_co
ntroller into state active
[INFO] [ros2-11]: process has finished cleanly [pid 19340]
[ros2-10] Sucessfully loaded controller arm_joint_trajectory_c
ontroller into state active
[INFO] [ros2-12]: process has finished cleanly [pid 19342]
[INFO] [ros2-13]: process has finished cleanly [pid 19344]
[INFO] [ros2-14]: process has finished cleanly [pid 19346]
[INFO] [ros2-10]: process has finished cleanly [pid 19338]
```



space_robots : docker — Konsole
File Edit View Bookmarks Settings Help
Gazebo
Image dis...
Copilot monitor
Violation

```
[INFO] [1705565733.943088967] [ros_gz_bridge]: Creating GZ->RO
S Bridge: [/copilot/prop1 (gz.msgs.Empty) -> /copilot/prop1 (s
td_msgs/msg/Empty)] (Lazy 0)
[INFO] [1705565733.944043172] [ros_gz_bridge]: Creating ROS->G
Z Bridge: [/copilot/prop1 (std_msgs/msg/Empty) -> /copilot/pro
p1 (gz.msgs.Empty)] (Lazy 0)
[INFO] [1705565793.303894932] [ros_gz_bridge]: Passing message
 from ROS std_msgs/msg/Empty to Gazebo gz.msgs.Empty (showing
msg only once per type)

   1 sh
response:
std_srvs.srv.Empty_Response()

# ros2 service call /move_stop std_srvs/srv/Empty
requester: making request: std_srvs.srv.Empty_Request()

response:
std_srvs.srv.Empty_Response()

# 
   2 sh
# source install/setup.bash
# ros2 run copilot copilot
[WARN] [1705565793.301353079] [rcl.logging_rosout]: Publisher
already registered for provided node name. If this is due to m
ultiple nodes with the same name then all logs for that logger
 name will go out over the existing publisher. As soon as any
node with that name is destructed it will unregister the publi
sher, preventing any further logs for that name from being pub
lished on the rosout topic.

   4 sh
ler into state active
[ros2-14] Sucessfully loaded controller wheel_tree_position_co
ntroller into state active
[INFO] [ros2-11]: process has finished cleanly [pid 19340]
[ros2-10] Sucessfully loaded controller arm_joint_trajectory_c
ontroller into state active
[INFO] [ros2-12]: process has finished cleanly [pid 19342]
[INFO] [ros2-13]: process has finished cleanly [pid 19344]
[INFO] [ros2-14]: process has finished cleanly [pid 19346]
[INFO] [ros2-10]: process has finished cleanly [pid 19338]
```

https://github.com/Copilot-Language/copilot/

spaceROS

Lunar Command-and-Control Interoperability Project (LuCCI)

We use Space ROS to evaluate different bridges between cFS and ROS. We have multiple simulations using both the JAXA and Traclabs bridges.

We have communicated feedback on cFS-ROS bridges to upstream maintainers.

Will be seeking integration with Isaac SIM in FY25.

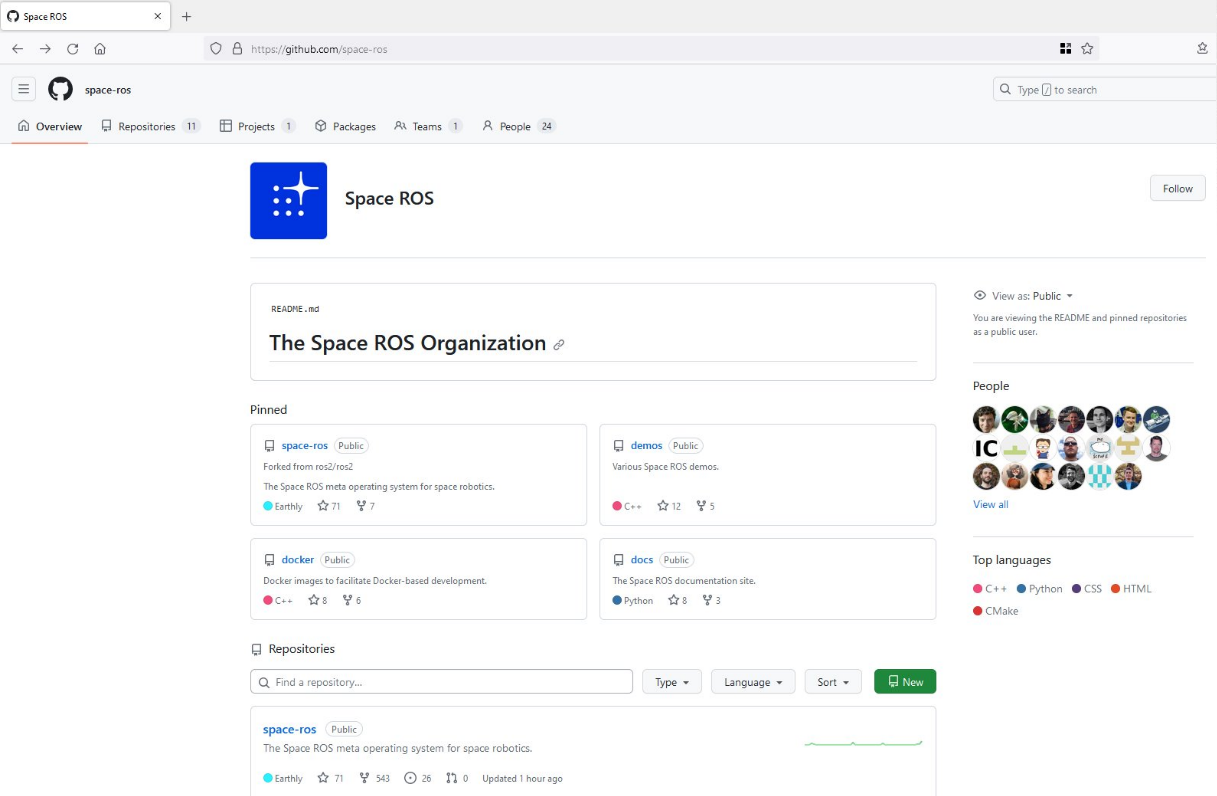
We have carried out a survey of flight and robotics software frameworks used in different missions (paper available), and we are conducting a trade study on communication mechanisms between flight and robotics software stacks (ongoing).

Our wish for Space Grade Linux

- Our deployment is a docker image based on docker, but Space ROS can be used on other Linux system.
- A docker image based on Space Grade Linux could be used as the base for future Space ROS images, immediately bringing a production-ready stack to Space Grade Linux with minimal effort.
- The same ideas, systems and techniques we use to make Space ROS systems safe could be used for Space Grade Linux, and systems based on Space Grade Linux.

Space ROS Github Organization

https://github.com/space-ros



Space ROS

README.md

The Space ROS Organization

You are viewing the README and pinned repositories as a public user.

Pinned

space-ros Public
Forked from ros2/ros2
The Space ROS meta operating system for space robotics.
● Earthly ☆ 71 ⑂ 7

demos Public
Various Space ROS demos.
● C++ ☆ 12 ⑂ 5

docker Public
Docker images to facilitate Docker-based development.
● C++ ☆ 8 ⑂ 6

docs Public
The Space ROS documentation site.
● Python ☆ 8 ⑂ 3

People

View all

Top languages

● C++ ● Python ● CSS ● HTML ● CMake

Repositories

space-ros Public
The Space ROS meta operating system for space robotics.
● Earthly ☆ 71 ⑂ 543 ⊙ 26 ⇅ 0 Updated 1 hour ago

spaceROS

Ivan Perez
KBR @ NASA Ames Research Center
ivan.perezdominguez@nasa.gov

Thanks to:

Tommy Madsen, Matt Hansen, Michael Jeronimo, Brian Kempa, Blazej Fiderek, Ana Huaman, Dharini Dutia, Geoffrey Biggs, Ezra Brooks, Alexey Simonov, Dave Coleman, all past and present project partners, and everyone who has contributed!