Open Robotics 2020:

ROS2 Eloquent and Ignition Citadel

Steve Peters
Open Robotics
Who we are

We create open software and hardware platforms for robotics. We use those platforms to solve important problems and we help others to do the same.
Where we are
Our products

Open source robotics developer tools

**ROS**: Robot application SDK

**Gazebo / Ignition**: Robot simulator

- Develop and test in simulation
- Deploy same software to robots
- Focus on differentiating capabilities
Milestone 3 Complete: PR2 Betas Ready and ROS 1.0

Submitted by admin on Fri, 01/22/2010 - 17:42

Today, we finished our third milestone! Simply put, ROS has reached 1.0 status. We also recently unveiled the PR2 Beta robots and the PR2 Beta Program, which will distribute approximately 10 PR2 robots at no cost.

Of course, it’s a lot more than that. Since work began on Milestone 3, there are now:

- 203 ROS software tutorials
- 29 ROS Stacks at 1.0 status, which contain a total of 186 ROS Packages
- 21 Completed Use Cases, requiring well over one hundred user studies
(some of the) ROS-based products available today
September 2019: (at least) $143.5M in investment

Embark: $70M Series B

Simbe: $26M Series A

Farmwise: $14.5M Series A

Built: $33M Series B
ROS Melodic: LTS release
(May 23 2018 - May 2023)

Python 2.7 will retire in...

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>22</td>
<td>5</td>
<td>47</td>
</tr>
<tr>
<td>Years</td>
<td>Months</td>
<td>Days</td>
<td>Hours</td>
<td>Minutes</td>
</tr>
</tbody>
</table>

What's all this, then?

Python 2.7 will not be maintained past 2020. Originally, there was no official date. Recently, that date has been updated to January 1, 2020. This clock has been

https://pythonclock.org/
ROS Noetic: planned LTS release
(May 2020 - May 2025)

Full migration to python3

ros-melodic-rospy depends on
- python-numpy
- python-yaml
- ...

ros-noetic-rospy will depend on
- python3-numpy
- python3-yaml
- ...

The last ROS 1 distribution?
The next generation
ROS 2: Goals

1. Quality of design & implementation
2. System reliability
3. Real-time control & deterministic execution
4. Validation, verification, and certification
5. Flexibility in communication
6. Support for small embedded systems
**Embedded Systems**

- Lighter weight communication system
- More modular design

**DDS-XRCE demonstration for the Renesas RX65N MCU**

**Demonstration overview**

This demonstration implements a PROSima Micro-XRCE-DDS Client as a DDS-XRCE implementation to RX65N MCU. It was described in the Renesas news release. The software can send/receive DDS "std_msgs/String" to/from Micro-XRCE-DDS Agent. It is implemented at the top of the AWS FreeRTOS and has room for other embedded applications to run. The below is the RX65N evaluation board line-up and GR-ROSE was used for this demonstration.

- GR-ROSE (will be available from Core Corporation)
Mobility

- New navigation stack
- More options for inter-robot communication

“We’ve had some aspects of ROS2 running in production-critical systems for nearly 3 months now” - Ryan Gariepy, Oct 2018
Automotive

- Deterministic execution
- Safety & certification
ROS 2 Technical Steering Committee (TSC)

- Manage roadmap
- Contribute development effort
- Set developer policies
ROS 2 Dashing: First LTS release
(May 31 2019 - May 2021)

1. Actions: Python support & command line tool
2. Composable nodes (nodelet performance without rewriting)
3. MoveIt 2 Alpha
4. Testing: QA, performance, security
5. Diagnostics
6. armhf support at Tier 2
ROS 2 Eloquent: Latest release (Nov 23 2019 - Nov 2020)

1. Launch files in XML/YAML
2. RVIZ interactive markers
3. ros2doctor CLI
4. Launch-based testing
5. RMW interface to iceoryx for zero-copy IPC
6. Cyclone DDS new Tier-2 RMW implementation
Recent features

- Composable nodes
- Potential for zero-copy comms (in C++)

```cpp
// Create a Talker class that subclasses the generic rclcpp::Node base class.
// The main function below will instantiate the class as a ROS node.
class Talker : public rclcpp::Node
...
RCLCPP_COMPONENTS_REGISTER_NODE(demo_nodes_cpp::Talker)
```
ros1_bridge

https://github.com/mabelzhang/ros1_bridge_sandbox
Gazebo simulator: Organizing principles

*Provide the best software stand-in for a physical robot*

1. Physics
2. Sensing
3. Extension
4. Interface with ROS
Physics

- No single engine is best for all situations
- Common API atop multiple physics engines
- Choose engine at runtime
- Maximal and reduced coordinate approaches
- Allows simple engines, e.g., kinematics-only
SENSING

- Parameterizable models of common sensor types
- Parameterizable models of common noise types
- Common API atop multiple rendering engines
- Export sensor data via middleware (e.g., ROS)
EXTENSION

• C++ plugin API allows any kind of extension
• Get and/or set the world between physics steps
• Add or extend sensors
• Interface with hardware input devices
• Fake interactions that are impractical to simulate
• Delegate interactions to other systems
Interface ROS and Gazebo

github.com/chapulina/roscon_gz_ros2
gazebo_ros_pkgs

github.com/chapulina/roscon_gz_ros2
gazebo_ros_pkgs

github.com/chapulina/roscon_gz_ros2
ROS topics/services

gazebo_ros_pkgs

gz topics / C++ API

github.com/chapulina/roscon_gz_ros2
Port gazebo_ros_pkgs to ROS 2.0 #512

j-rivero opened this issue on Nov 29, 2016 · 25 comments

j-rivero commented on Nov 29, 2016 · edited by chapulina

Interest into porting gazebo_ros_pkgs to ROS 2.0 is starting to appear within the ROS Community, this post tries to server a tracker for suggestion, status and discussions on how to do it.

Update 29 Nov 2016
The ROS team currently warns about implementations at this point since the API is still unstable an future changes could require a big amount of work to update the code. Even with API instability a possible suggestion to start the works could be some time after the Beta1 (release planned in the next weeks), late January if the feedback regarding to the Beta1 is good at that moment.

Update 2nd Jun 2018
The branch ros2 is ready in this repository to start merging changes related to port the code to ROS2.

Update 19 July 2018
Written by @ironmig
Gazebo versions

Gazebo 7: LTS Jan 2016 - Jan 2021
Gazebo 9: LTS Jan 2018 - Jan 2023
Gazebo 10: Jan 2019 - Jan 2021
Gazebo 11: LTS Jan 2020 - Jan 2025
MODULARITY

- Monolithic Gazebo decomposed into Ignition libraries
- Libraries can be reused in other applications
- Ignition Gazebo is just one particular composition
Linking to physics engines in gazebo9

Find packages during configuration
Link during build
Choose physics engine at runtime

Gazebo CMakeLists.txt:
```cmake
find_package(bullet)
...
if (bullet_FOUND)
target_link_libraries(
gazebo_physics ${bullet_LIBRARIES})
endif()
```

Bash prompt:
```
$ gazebo --verbose -e bullet
$ gazebo --verbose -e dart
[Err] Unregistered physics engine [dart]
```
ign-physics: plugins for different physics engines!

```cpp
using MinimumFeatureList = ignition::physics::FeatureList<
    // FreeGroup
    ignition::physics::FindFreeGroupFeature,
    ignition::physics::SetFreeGroupWorldPose,
    ignition::physics::FreeGroupFrameSemantics,
    ignition::physics::LinkFrameSemantics,
    ignition::physics::AddLinkExternalForceTorque,
    ...
    ignition::physics::sdf::ConstructSdfModel,
    ignition::physics::sdf::ConstructSdfVisual,
    ignition::physics::sdf::ConstructSdfWorld
>;

using EnginePtrType = ignition::physics::EnginePtr<
    ignition::physics::FeaturePolicy3d, MinimumFeatureList>;

ignition::plugin::Loader pl;
ignition::plugin::PluginPtr plugin = pl.Instantiate("ignition::physics::dartsim::Plugin");

EnginePtrType engine = ignition::physics::RequestEngine<
    ignition::physics::FeaturePolicy3d,
    PhysicsPrivate::MinimumFeatureList>::From(plugin);
```
ign-rendering
ign-gazebo
Ignition Blueprint
(31 May 2019)

1. Physically based rendering (PBR) materials
2. GUI tools for model placement
3. Payload-dependent battery model
4. New command line tools
5. Incremental level loading
6. Distributed simulation (work in progress)
Ignition Citadel
(December 2019)

1. Actors

2. Thermal Camera

3. Visual markers

4. SDFFormat frame semantics

<pose relative_to="parent_link">..."/pose>
Pose frame semantics in SDFormat 1.6

```xml
<sdf version="1.6">
  <model name="pendulum_with_base">
    <link name="base">
      <pose>0 0 0.3 0 0 0</pose>
    </link>
    <link name="pendulum">
      <pose>0 0.5 1.0 1.57 0 0</pose>
    </link>
    <joint name="joint" type="revolute">
      <parent>base</parent>
      <child>pendulum</child>
      <pose>0 0 0.5 0 0 0</pose>
      <axis>
        <xyz>1 0 0</xyz>
      </axis>
    </joint>
  </model>
</sdf>
```
Pose frame semantics in SDFormat 1.7

```
<sdf version="1.7">
  <model name="pendulum_with_base">
    <link name="base">
      <pose>0 0 0.3 0 0 0</pose>
    </link>
    <link name="pendulum">
      <pose relative_to="joint">
        0 0 -0.5 0 0 0
      </pose>
    </link>
    <joint name="joint" type="revolute">
      <parent>base</parent>
      <child>pendulum</child>
      <pose relative_to="base">
        0 0 0.73 1.57 0 0
      </pose>
      <axis>
        <xyz>1 0 0</xyz>
      </axis>
    </joint>
  </model>
</sdf>
```
topics

diagram:
- Multiple topics (represented by circles)
- ros_ign
- Bridge and envelope icon
- Additional topics (represented by circles)

topics
Investment generally results in acquiring an asset, also called an investment. If the asset is available at a price worth investing, it is normally expected either to generate income, or to appreciate in value, so that it can be sold at a higher price.