It Takes a Village to Build a Robot: Understanding the ROS Ecosystem

Carnegie Mellon University
Columbia University
In the City of New York

Sophia Kolak, Chris Timperley, Afsoon Afzal, Michael Hilton, Claire Le Goues
The ROS vision: Better Together!

✔ Code reuse

✔ Modularity

✔ Faster Development

✔ Easier Maintenance
Are we Living up to the Vision?
What happens in the wild?
In practice, Modularity is Hard.

33% of ROS bugs are Dependency Errors

74% of ROS users have failed to implement a Dependency

1. How ROS cares for Quality, ROSin, ROScon 2017
2. Estefo et al., The Robot Operating System: Package reuse and community dynamics
A Deeper Look at **Dependence**: Modeling the ROS Software Ecosystem.

**foo** is depended on twice

**foo** has three dependencies
Packages Usually Depend on \(~9\) Others.
But most packages are never depended on...

Is Depended on 2016, Kinetic

Is Depended on 2019, Kinetic
A Few Packages Do Most of the Work

~19% have one dependent package
~12% have two dependent packages
~61% of all kinetic packages are depended on globally less than 5 times
Dependence Inequality?

The top 1% of Packages are Depended on More than the bottom 99%.
Dependence Inequality?

The top 1% of Packages are Depended on More than the bottom 99%.

So What?
Power laws are common in OSS

“20% of nodes responsible for 80% of result”
In FOSS Ecosystems, Github metrics are often correlated with popularity.

Figure 5: Differences in ranks between popularity measures.
In ROS, Github Metrics Don’t Explain Popularity.
Being Maintained by OSRF Matters More.

Top 20, 2016

Top 20, 2019
Possible Explanations

- Of 230,000 ROS packages on Github, only 45,000 were uniquely named.
- 8 of the 20 most copied packages are never specified as an explicit dependency.
Possible Explanations

- No sophisticated search mechanism
- Many packages are not on ROS index or wiki

Users can’t find new packages
Possible Explanations

- Documentation is often minimal (average readme length is 19 lines)
- Packages rarely state the version they depend on (~1% of the time)
- Not maintained, some packages in ROSdistro were abandoned

Users can’t implement new packages
Suggestions:

- Teach others, document better
- Semantic Search Mechanism
- Explicitly State Dependencies
- Make versioning clear
- Lightweight Vetting process?
Thank You!

- Should we check quality of code in the ROS distro?
- How should the ecosystem evolve for ROS2?
- How did we gather our data?
How we structured our data

```xml
<package format="2">
  <name>actionlib</name>
  <author>Eitan Marder-Eppstein</author>
  <author>Vijay Pradeep</author>
  <author>Mikael Arguedas</author>

  <buildtool_depend version_gte="0.5.78">catkin</buildtool_depend>

  <build_depend>message_generation</build_depend>

  <depend>actionlib_msgs</depend>
  <depend>boost</depend>
  <depend>roscpp</depend>
  <depend>rospy</depend>
  <depend>rostest</depend>
  <depend>std_msgs</depend>

  <exec_depend>message_runtime</exec_depend>
  <exec_depend>python-wxtools</exec_depend>
  <exec_depend>roslib</exec_depend>
  <exec_depend>rostopic</exec_depend>

  <test_depend>rosnode</test_depend>
</package>
```
actionlib

<depend>actionlib_msgs</depend>
<depend>boost</depend>
<depend>roscpp</depend>
In mySQL...

<table>
<thead>
<tr>
<th>id</th>
<th>parent_id</th>
<th>child</th>
<th>dep_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>81059</td>
<td>21265</td>
<td>actionlib_msgs</td>
<td>depend</td>
</tr>
<tr>
<td>81060</td>
<td>21265</td>
<td>boost</td>
<td>depend</td>
</tr>
<tr>
<td>81061</td>
<td>21265</td>
<td>roscpp</td>
<td>depend</td>
</tr>
</tbody>
</table>

<depend>actionlib_msgs</depend>
<depend>boost</depend>
<depend>roscpp</depend>
<depend>actionlib_msgs</depend>
<depend>boost</depend>
<depend>roscpp</depend>

In mySQL...

<table>
<thead>
<tr>
<th>id</th>
<th>parent_id</th>
<th>child</th>
<th>dep_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>81059</td>
<td>21265</td>
<td>actionlib_msgs</td>
<td>depend</td>
</tr>
<tr>
<td>81060</td>
<td>21265</td>
<td>boost</td>
<td>depend</td>
</tr>
<tr>
<td>81061</td>
<td>21265</td>
<td>roscpp</td>
<td>depend</td>
</tr>
</tbody>
</table>
Are Top Packages Just Better?

Used Github data as a proxy for quality