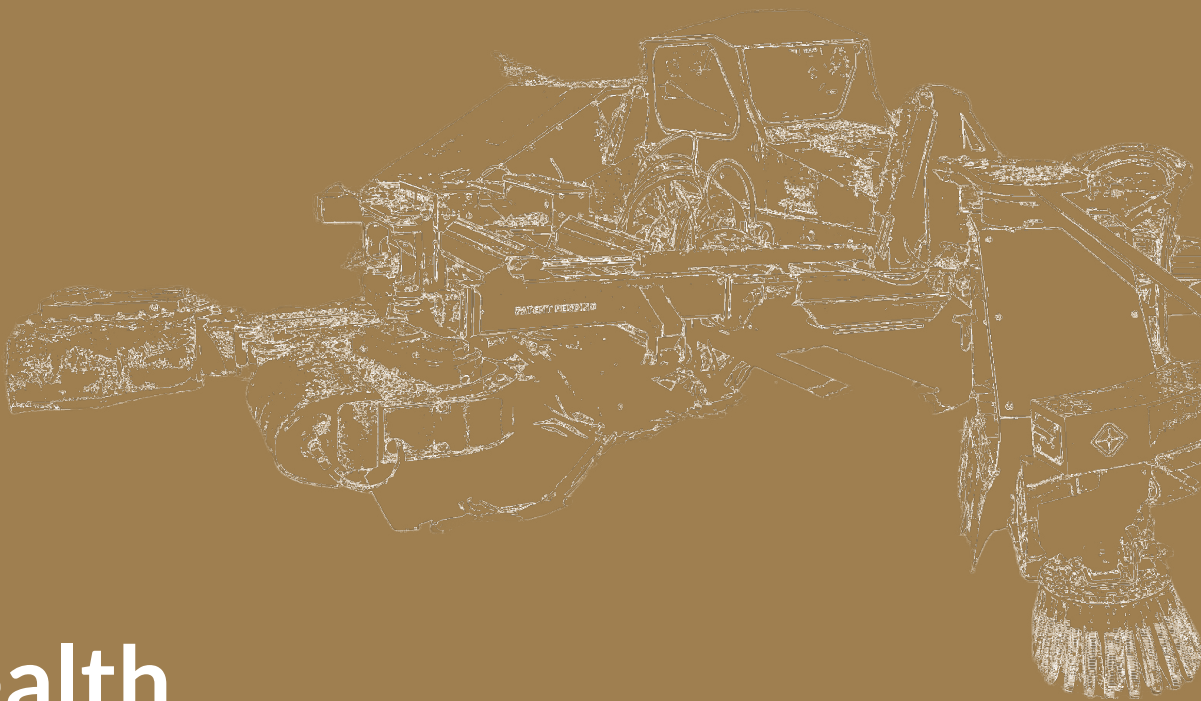


ROS Robot  
Health Monitoring:  
a Bonsai approach

bonsai

Emerson Knapp  
Staff Robotics Engineer





# Application Health

Robo, why no go?

# What is “robot application”?

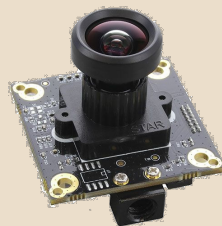
## → Not *just* ROS

- ◆ Configured host computer
- ◆ Various system services
- ◆ Connections to internet
- ◆ Connections to hardware
- ◆ ROS

## → What's a ROS application?

- ◆ Connected, communicating graph of special-purpose programs

## → Let's take a holistic view

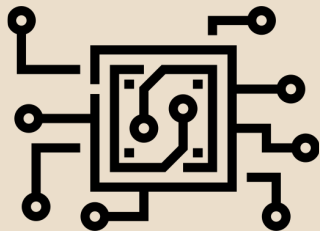


[ ● ◀ ] systemd



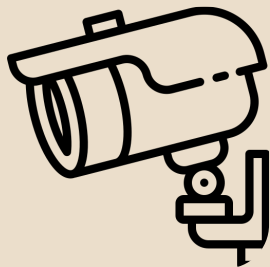
# What is “health”?

## Our Categories



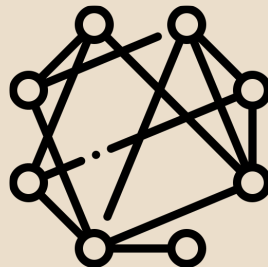
### System Diagnostics

Is system resource usage within appropriate bounds?



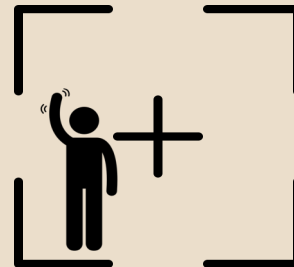
### Hardware Diagnostics

Are expected hardware devices connected and communicating?



### Application Level / Graph Diagnostics

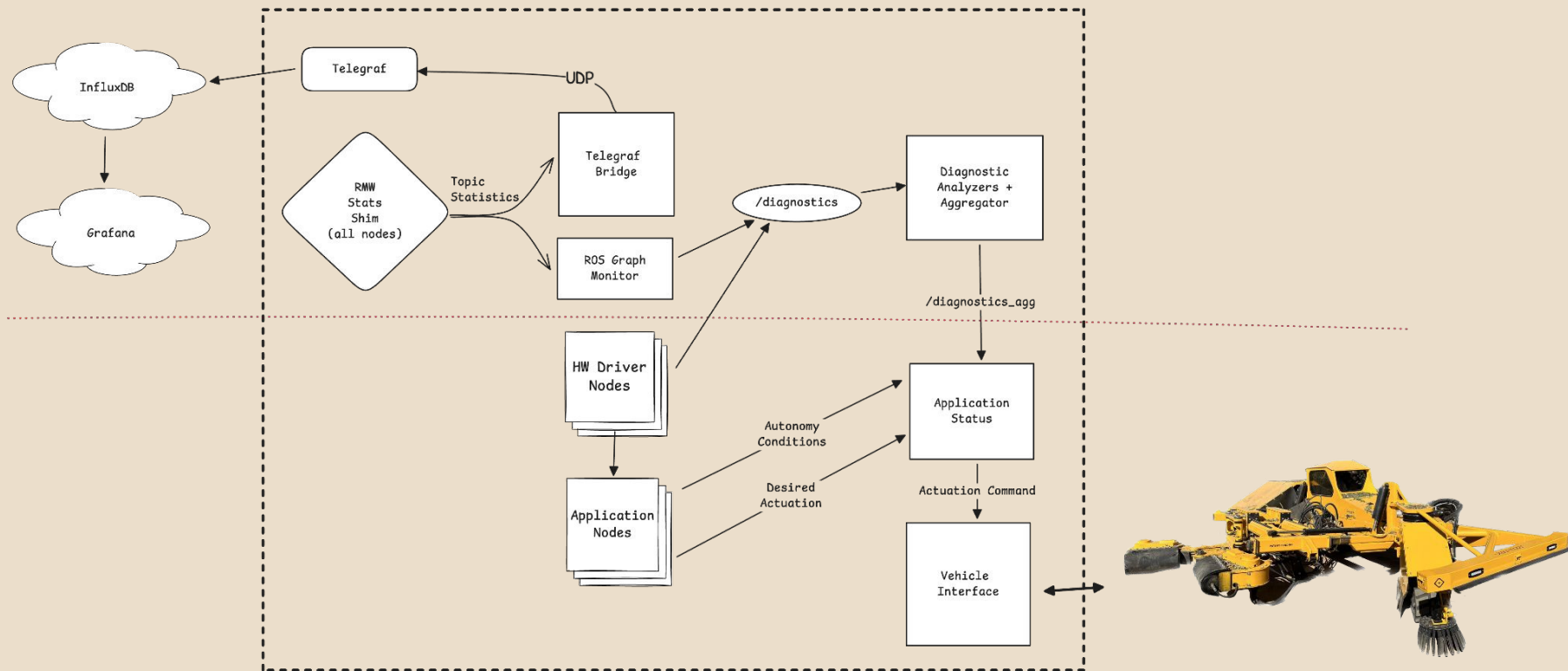
Is the ROS graph behaving?



### Logical Autonomy Conditions

Health seems fine, but something is preventing us from continuing.

## ■ Bonsai Health Architecture





# Today let's focus on:

## → Developer perspective

We're devs, trying to debug our bot

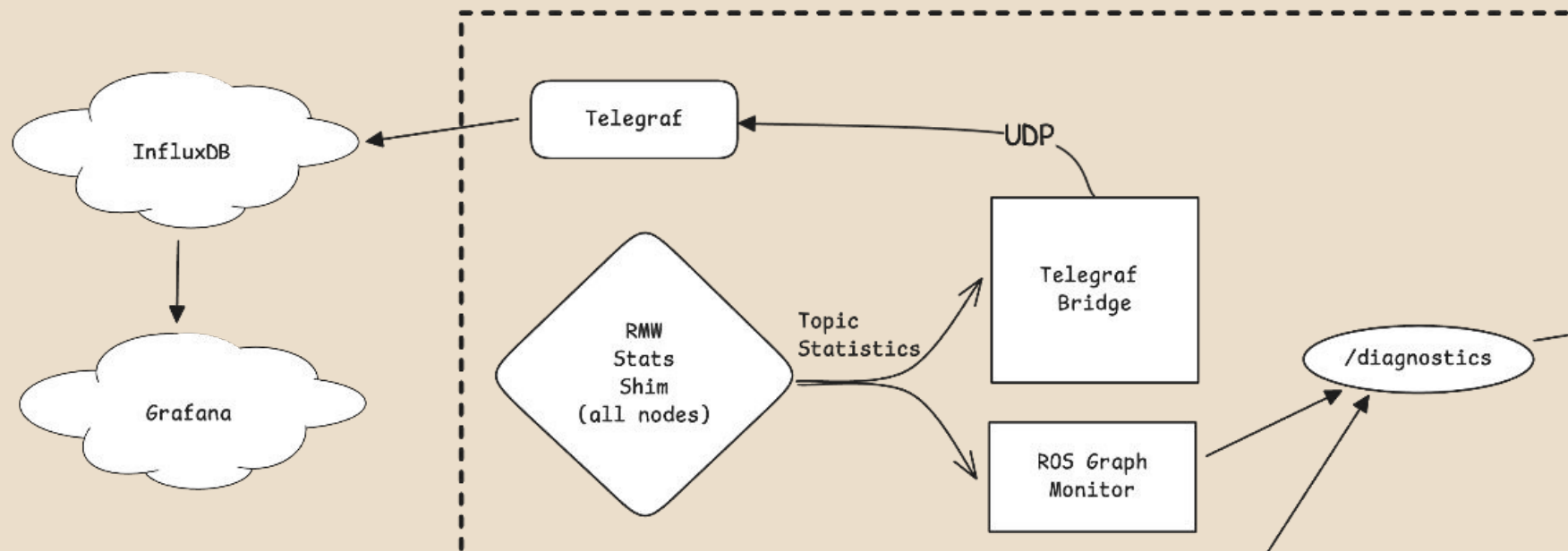
BUT, Some info also valuable to surface to user

## → Universal Parts: Host Metrics & Graph Monitoring

ROS Diagnostics covered already &

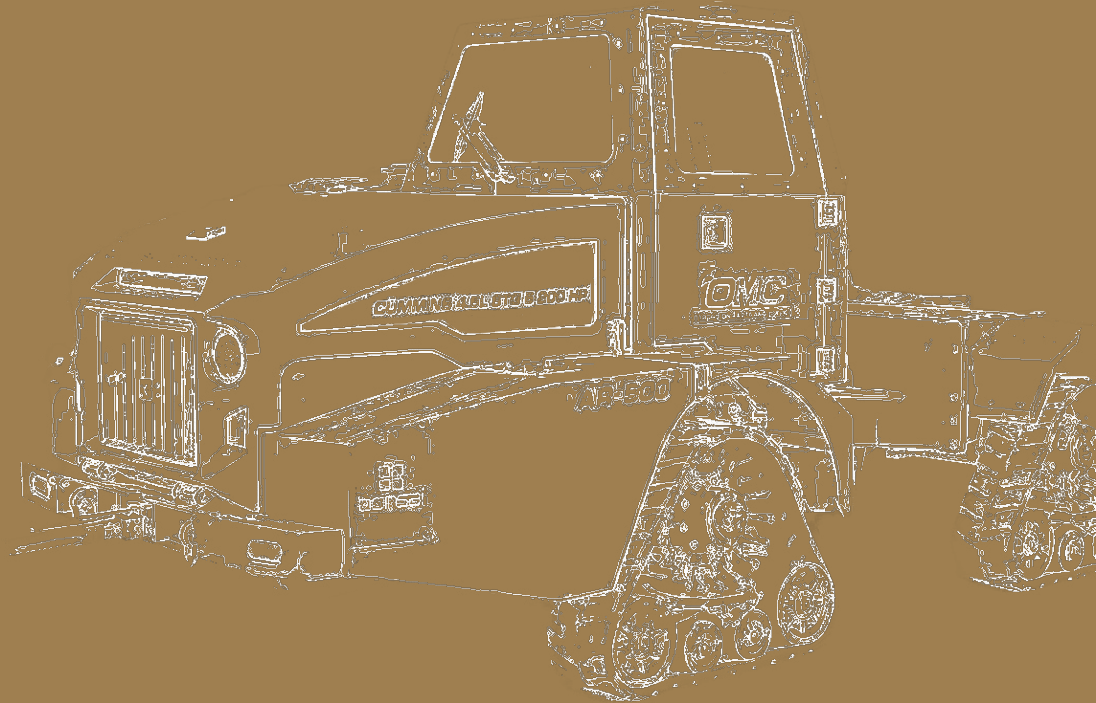
Hardware & Autonomy checks more specific to your application

## ■ Bonsai Health Architecture



# The TIG Stack

System metrics - one approach





# TIG

## → Telegraf

Host-side metrics collector with lots of built-in plugins, and easy to extend

## → InfluxDB

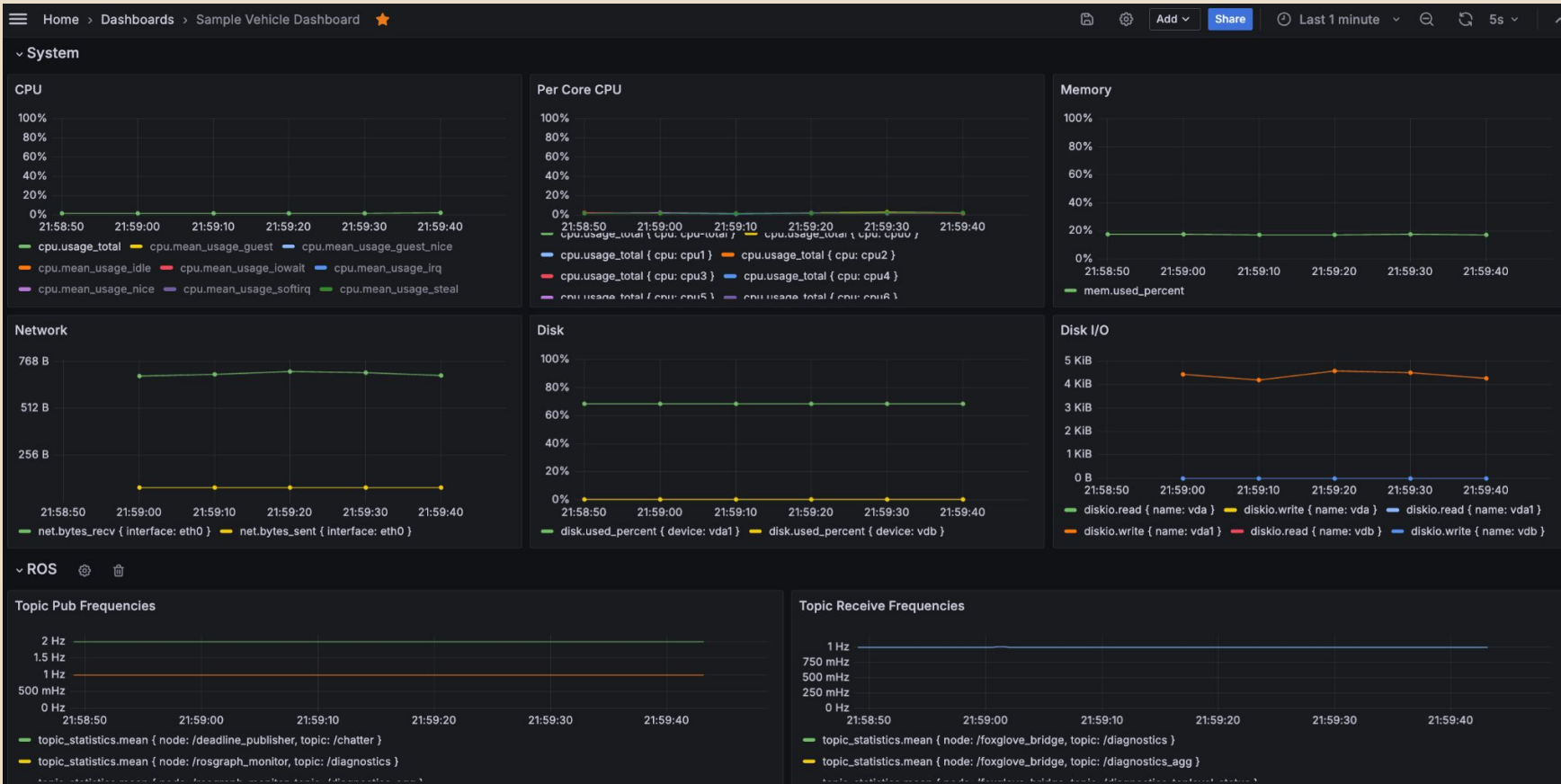
Performant Time Series Database, integrates by design with Telegraf

## → Grafana

Open source easy-to-use dashboarding platform

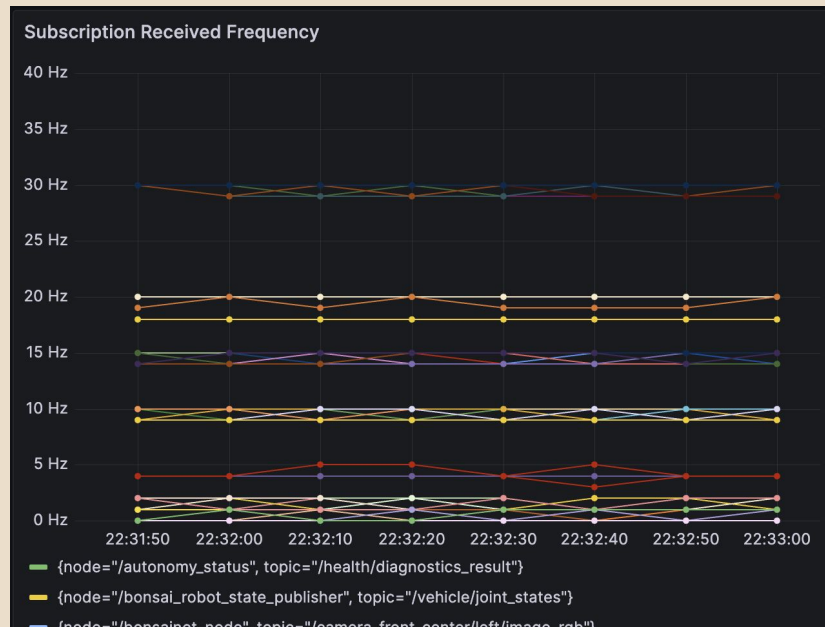


## System Metrics - TIG



# Telegraf Publisher Node

- Simple ROS 2 Node (Python)
- Sends UDP metrics -> Telegraf
- /topic\_statistics in
- Room to improve: want to make this bidirectional, pluggable to extend





# ROS Application Monitoring

Turtle graphs all the way down

# Graph Problems

## *Node Gone*

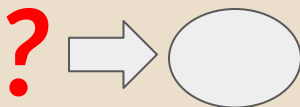
A program isn't even running



## *Discontinuities*

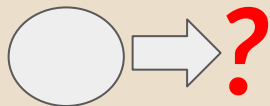
### **Dead Sink**

No publishers



### **Leaf topic**

No subscribers



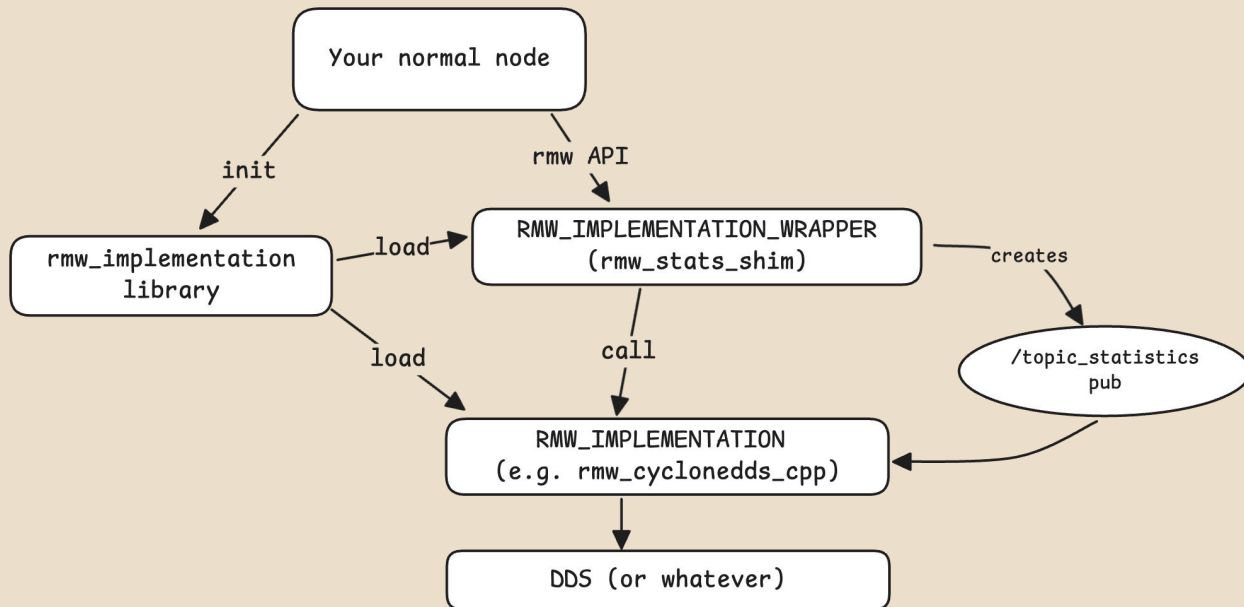
## *Message Rate*

Data not flowing correctly.



# Topic Statistics!

Core feature not usable (my bad, sorry). Here's what we're doing:





# Topic Statistics Monitoring

All you have to do:

- Set Deadline QoS on Publishers (that's it, no callbacks, no subs)
- Python:  
`rclpy.qos.QoSProfile(depth=X, deadline=Duration(seconds=0.2)))`
- C++:  
`rclcpp::QoS(rclcpp::KeepLast(X)).deadline(std::chrono::milliseconds(200))`

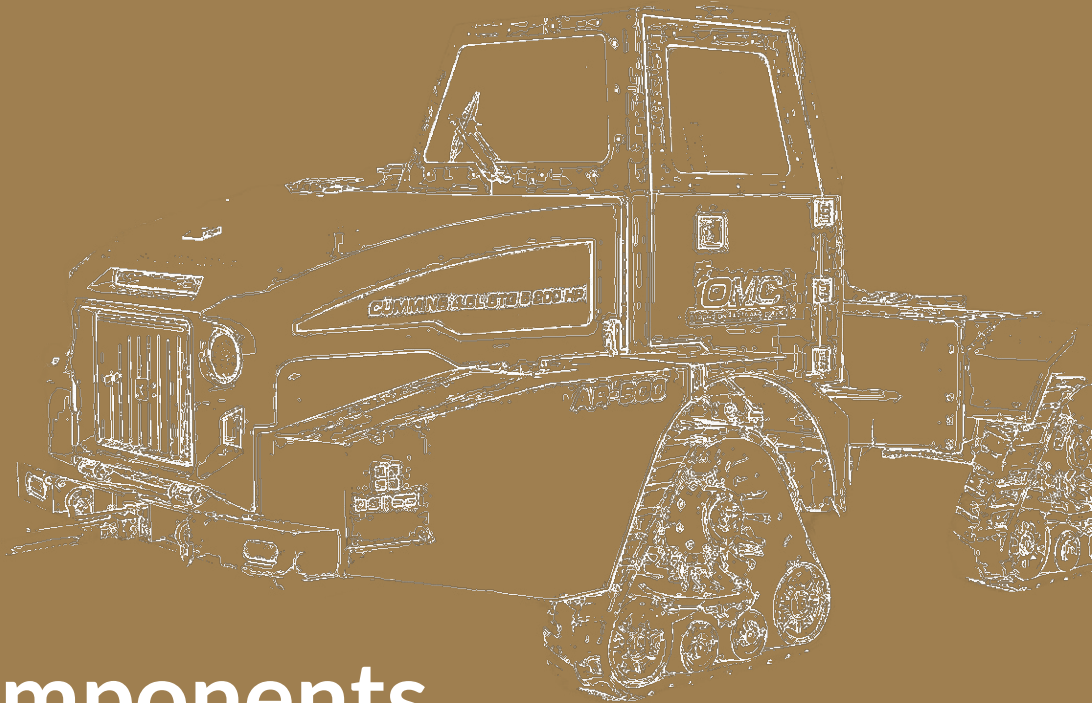
# The GraphMonitor

- Single class, uses NodeGraphInterface - detects graph problems to Diagnostics

The screenshot displays the ROS2 diagnostic tool interface, showing the health status of the RosGraph. The interface is divided into three main panels:

- Left Panel (Diagnostics - Summary (ROS)):** Shows a list of diagnostic messages. The top message is "health: Rosgraph::Continuity::/ chatter Leaf topic (No subscriptions)". Below it, there are several other messages, including "health: Rosgraph::PublishFrequency::/ chatter Publisher sending slower than promised deadline".
- Middle Panel (Diagnostics - Summary (ROS)):** Shows a summary of the diagnostic messages. The top message is "/Health/RosGraph Warning". Below it, there are several other messages, including "/Health/RosGraph/PublishFrequency Required publish frequencies not being met".
- Right Panel (/diagnostics\_toplevel\_status):** Shows the top-level status of the diagnostics. The status is "Warning" (level 1). The message is "Warning". The hardware\_id is empty. The values section shows a list of diagnostic messages, including "/Health/RosGraph/PublishFrequency: Required publish frequencies not being met".

The overall status is "WARN", indicated by a yellow circle with a white exclamation mark and the word "WARN" in yellow text.



# Open Sourced Components

And a sample application

## Health Monitoring Packages

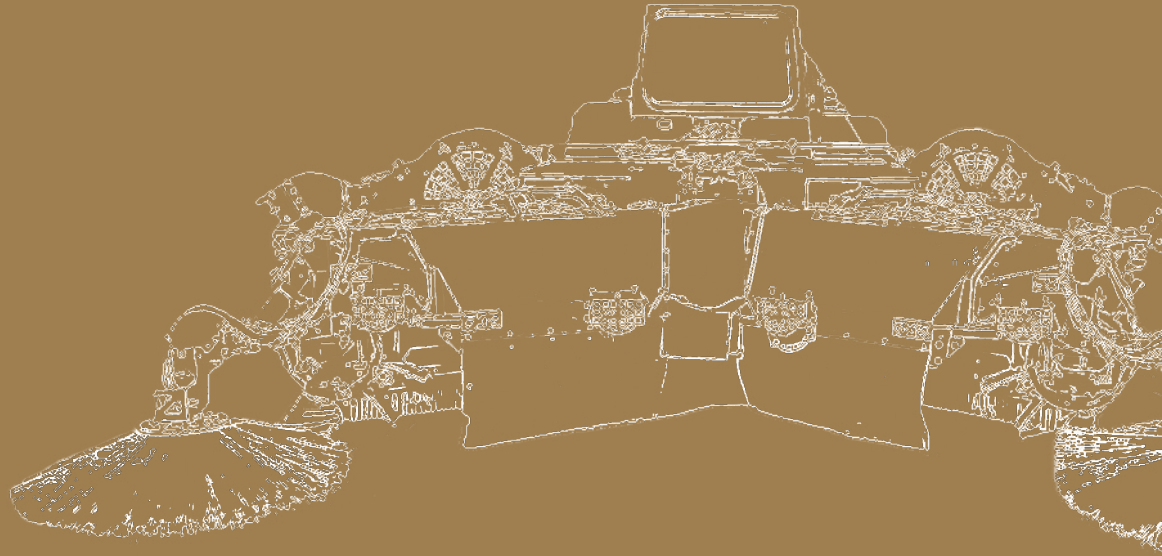
<https://bit.ly/ros-health-components>

- Real component packages that we are using internally:
- Patched `rmw_implementation`
- `rmw_stats_shim` : RMW wrapper to calculate and publish statistics
- `rosgraph_monitor`: ROS Graph Monitor & Diagnostic Analyzer
  - I'm not using diagnostics aggregator properly, come help!
- `telegraf_bridge`: just publishing `/topic_statistics` to Telegraf for now

# Sample Application

<https://bit.ly/ros-health-app>

- Provides VCS workspace.repos file for colcon workspace
- Entrypoint launchfile `sample.launch`
- `docker-compose` local full TIG stack with sample dashboard



# Practical Diagnosis

Four case studies of problems solved in the field



## ■ Case Study 1

diagnostics

Diagnostics – Summary (ROS)

OK Filter

health: Rosgraph::Continuity::/ chatter Dead sink: cleared. Topic now has publisher(s)

health: Rosgraph::Continuity::/diagnostics Leaf topic: cleared. Topic now has subscriber(s)

health: Rosgraph::Continuity::/diagnostics\_agg Leaf topic: cleared. Topic now has subscriber(s)

health: Rosgraph::Continuity::/diagnostics\_toplevel\_status Leaf topic: cleared. Topic now has subscriber(s)

health: Rosgraph::Continuity::/whatever Leaf topic (No subscriptions)

health: Rosgraph::Node::/robot\_state\_publisher Required node missing: /robot\_state\_publisher

health: Rosgraph::Node::/roscpp\_recorder Required node missing: /roscpp\_recorder

health: Rosgraph::Node::/rowline\_follower Required node missing: /rowline\_follower

health: Rosgraph::Node::/upload\_manager Required node missing: /upload\_manager

health: Rosgraph::PublishFrequency::/ chatter Publisher sending slower than promised deadline

health: Rosgraph::Node::/data\_manager Node OK: /data\_manager

health: Rosgraph::Node::/deadline\_publisher Node OK: /deadline\_publisher

Diagnostics – Summary (ROS)

OK Filter

/Health/RosGraph Error

/Health/RosGraph/Nodes Required nodes missing

/Health/RosGraph/PublishFrequency Required publi...

/diagnostics\_toplevel\_status

diagnostic\_msgs/msg/DiagnosticStatus @ 1729677837.157000000 sec

level 2 (ERROR)

name "/Health/RosGraph"

message "Error"

hardware\_id ""

values [] 2 items

0 /Health/RosGraph/Nodes: Required nodes missing

key "/Health/RosGraph/Nodes"

value "Required nodes missing"

1 /Health/RosGraph/PublishFrequency: Required publish frequencies not being met

key "/Health/RosGraph/PublishFrequency"

value "Required publish frequencies not being met"

health: Rosgraph::Node::/upload\_manager

health: Rosgraph::Node::/upload\_manager

health: Rosgraph::Node::/upload\_manager

Required node missing: /upload\_manager

/Health/RosGraph/Nodes

/Health/RosGraph/Nodes

/Health/RosGraph/Nodes

Required nodes missing


/robot\_state\_publisher Required node missing: /robot\_state\_publisher

/roscpp\_recorder Required node missing: /roscpp\_recorder

/rowline\_follower Required node missing: /rowline\_follower

/upload\_manager Required node missing: /upload\_manager

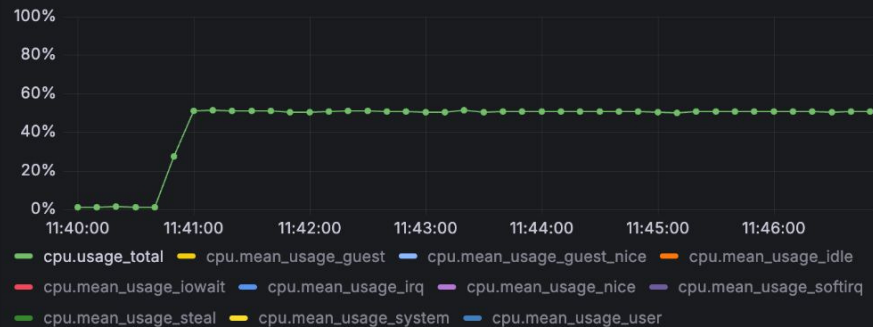
/diagnostics\_toplevel\_status.level

 ERROR

## ■ Case Study 1

### ~ System

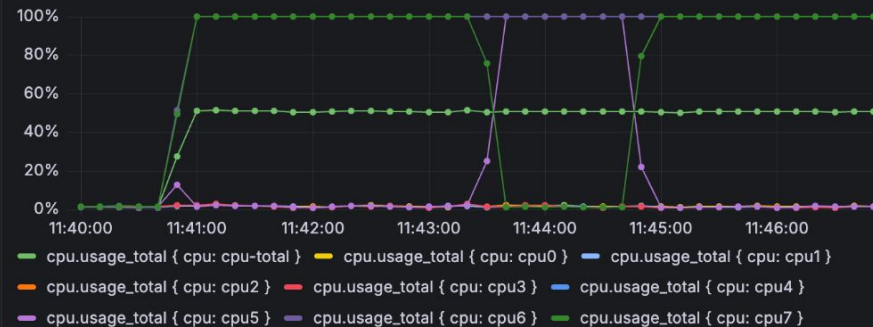
#### CPU



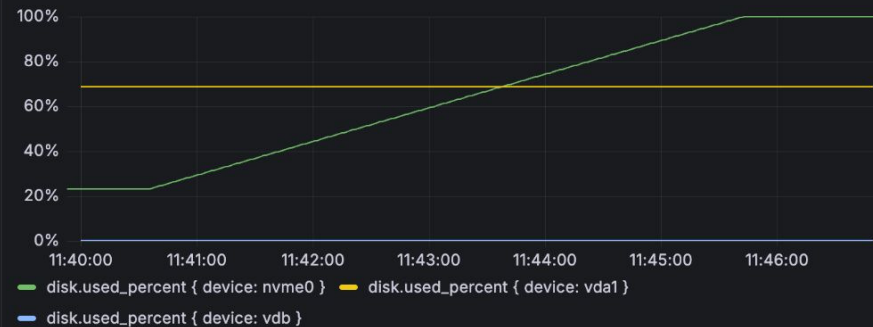
#### Memory




#### Per Core CPU



#### Disk



## ■ Case Study 2

diagnostics <span>×</span> Enter tab name	Diagnostics – Summary (ROS) <span>⚙️</span> <span>⋮</span>	Diagnostics – Summary (ROS) <span>⚙️</span> <span>⋮</span> <span>📄</span> <span>×</span> /diagnostics_toplevel_status <span>⚙️</span> <span>⋮</span>
<div><span>OK</span> <span>▼</span> <input type="text" value="Filter"/></div> <div><p>health: Rosgraph::Continuity::/camera_front_center/image_rgb <span>Leaf topic (No subscriptions)</span></p><p>health: Rosgraph::Continuity::/camera_front_center/image_rgb <span>Dead sink (No publishers)</span></p><p>health: Rosgraph::Node::/camera_front_ceter <span>Node OK: /camera_front_ceter</span></p><p>health: Rosgraph::Node::/tree_detector <span>Node OK: /tree_detector</span></p><p>health: Rosgraph::PublishFrequency::/chatter <span>Publisher send frequency acceptable</span></p></div>	<div><span>OK</span> <span>▼</span> <input type="text" value="Filter"/></div> <div><p>/Health/RosGraph <span>OK</span></p><p>/Health/RosGraph/Nodes <span>OK</span></p><p>/Health/RosGraph/PublishFrequency <span>OK</span></p></div>	<div>diagnostic_msgs/msg/DiagnosticStatus @ 1729678717.647000000 sec <span>📄</span></div> <div><p>level 0 (OK)</p><p>name "/Health/RosGraph"</p><p>message "OK"</p><p>hardware_id ""</p><p>values <span>[]</span> 0 items</p></div>
<div>health: Rosgraph::Continuity::/camera_front_center/image_rgb <span>⚙️</span> <span>⋮</span></div> <div><div>health: Rosgraph::Continuity::/camera_front_center/image_rgb <span>▼</span></div><div><p><b>health: Rosgraph::Continuity::/camera_front_center/image_rgb</b></p><p><b>Dead sink (No publishers)</b></p></div></div>	<div>/Health/RosGraph/Nodes <span>⚙️</span> <span>⋮</span></div> <div><div>/Health/RosGraph/Nodes <span>▼</span></div><div><p><b>/Health/RosGraph/Nodes</b></p><p><b>OK</b></p></div></div>	<div>/diagnostics_toplevel_status.level <span>⚙️</span> <span>⋮</span></div> <div> <b>OK</b></div>

## ■ Case Study 3

diagnostics

Enter tab name

Diagnostics – Summary (ROS)

OK Filter

health: Rosgraph::PublishFrequency::/camera\_front\_center/image\_rgb Publisher sending slower...

health: Rosgraph::PublishFrequency::/chatter Publisher sending slower than promised deadline

health: Rosgraph::Node::/camera\_front\_ceter Node OK: /camera\_front\_ceter

health: Rosgraph::Node::/tree\_detector Node OK: /tree\_detector

health: Rosgraph::PublishFrequency::/camera\_front\_center/image\_rgb

health: Rosgraph::PublishFrequency::/camera\_front\_center/image\_rgb

health: Rosgraph::PublishFrequency::/camera\_front\_center/image\_rgb

Publisher sending slower than promised deadline

Diagnostics – Summary (ROS)

OK Filter

/Health/RosGraph Warning

/Health/RosGraph/PublishFrequency Requir...

/Health/RosGraph/Nodes OK

/Health/RosGraph/Nodes

/Health/RosGraph/Nodes

/Health/RosGraph/Nodes

OK

/diagnostics\_toplevel\_status

diagnostic\_msgs/msg/DiagnosticStatus @ 1729678831.652000000 sec

level 1 (WARN)

name "/Health/RosGraph"

message "Warning"

hardware\_id ""


values 1 item

0 /Health/RosGraph/PublishFrequency: Required publish frequencies not being met

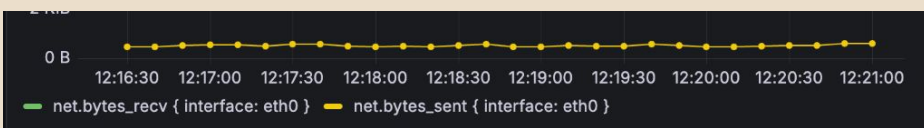
key "/Health/RosGraph/PublishFrequency"

value "Required publish frequencies not being met"

/diagnostics\_toplevel\_status.level

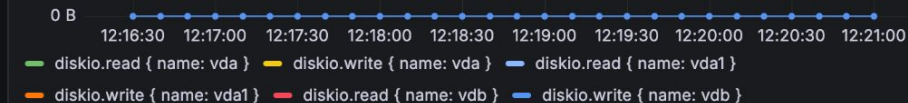
 WARN

## ■ Case Study 3

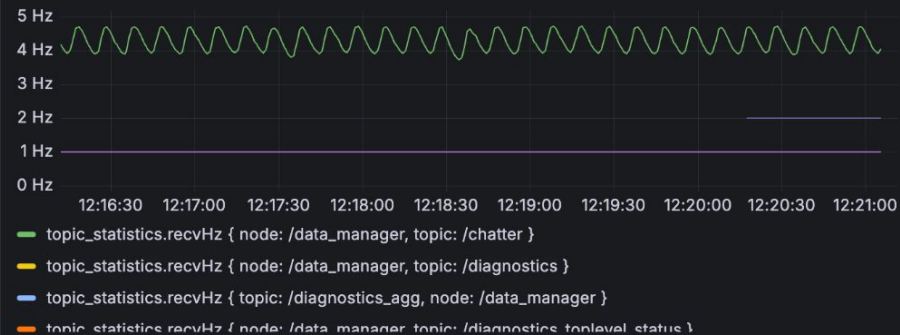


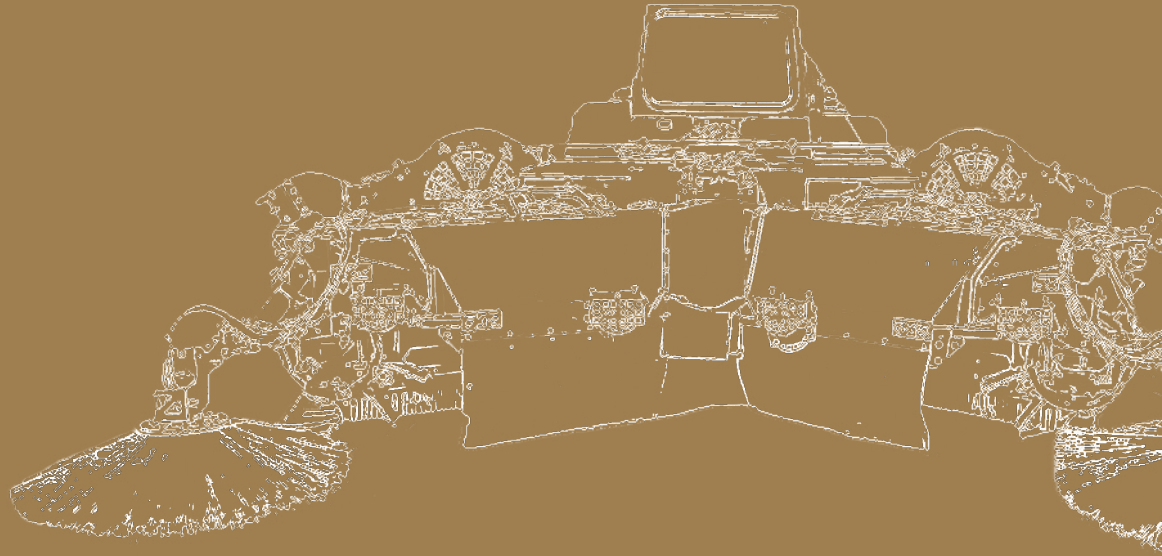
▼ ROS ⚙️ 🗑️

### Topic Pub Frequencies



### Topic Receive Frequencies





# Conclusion

Observability saves the day



## You didn't need the resident expert!

- Real time observability
- Declared/programmed system expectations
- Automatically monitored
- Anybody on the team can know where the problem is
- This is a way you scale your one-rockstar show to a larger team, and make that rockstar more productive to boot

## Thank you! Questions?

- Lots more to do
- Please come chat
- Also, please try the code and interact on Github
  - [https://github.com/BonsaiRobotics/roscon2024\\_health\\_sample](https://github.com/BonsaiRobotics/roscon2024_health_sample)
- Also, we're hiring
  - [bonsairobotics.ai/careers](https://bonsairobotics.ai/careers)