Scaling Open-RMF

Test Bench → Lab → Controlled Environment → Production

Akash Vibhute, Xi Yu Oh
Intrinsic

ROSCon 2024

Open-RMF

Agenda

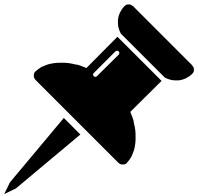
01	Getting started with Open-RMF	2 min
02	Testing with hardware	4 min
03	First run onsite	15 min
04	Calling it in production	4 min
05	Q&A	5 min

Open-RMF

1.

Getting started with Open-RMF

Let’s start simple

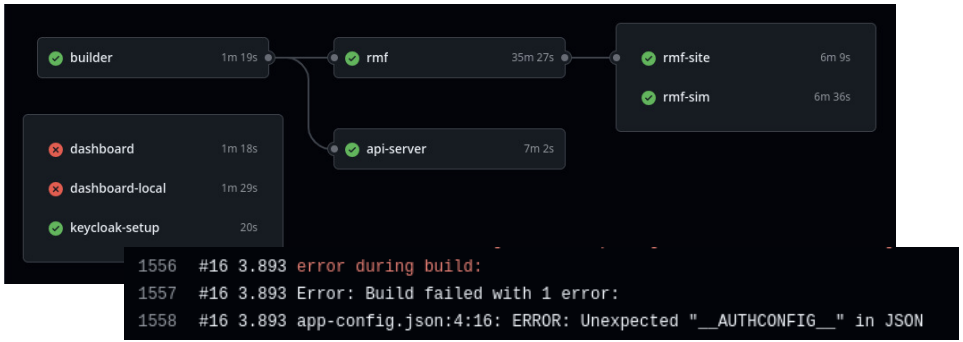


Getting started

github.com/open-rmf/rmf_deployment_template

- `docker-compose` for testing
- k8s for deployment
- CI template available!

The earlier you catch defects, the cheaper they are to fix.



Whoops!

Open-RMF

2.

Testing with hardware

Connect to real hardware where possible,
simulate/mock the rest

Open-RMF

Simulation

No hardware involved.
We can conduct repetitive testing using a fraction of the time for debugging.

Lab

Integrating RMF with real robots, putting the implemented fleet adapter to test.

Controlled environment

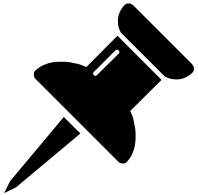
Running RMF at the actual site, discovering new limitations in the space, deployment network and building infrastructure we're dealing with.

Production

Co-exist with human staff along with scheduled building/robot maintenance, fire alarm evacuation exercises, etc.

Dev, Test

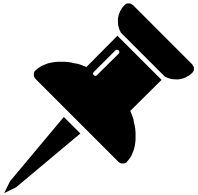
Open-RMF



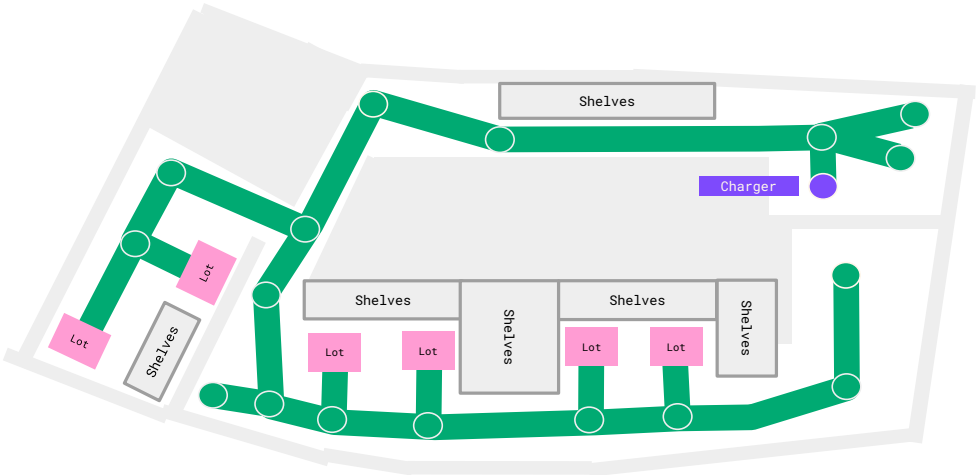
Testing with hardware

For effective **Open-RMF navigation graph** testing

- Add lanes depicting the deployment environment in your test traffic map
- Add “unplanned” obstacles to the environment
- Include Docking/Charging in test plan



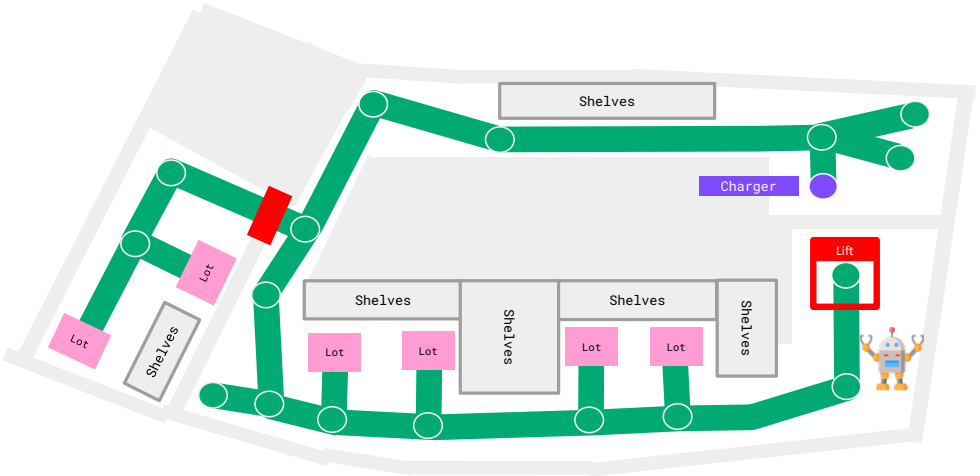
Testing with hardware



What's wrong with this diagram?

Open-RMF

Testing with hardware



Add doors, lifts to your testing.

Try interrupting the robot by introducing humans.

Open-RMF

Testing with hardware

Strategies when **simple hardware (eg. doors, beacons/alarms)** are unavailable

- Connect to a simulated device to publish response messages on your **real** topic

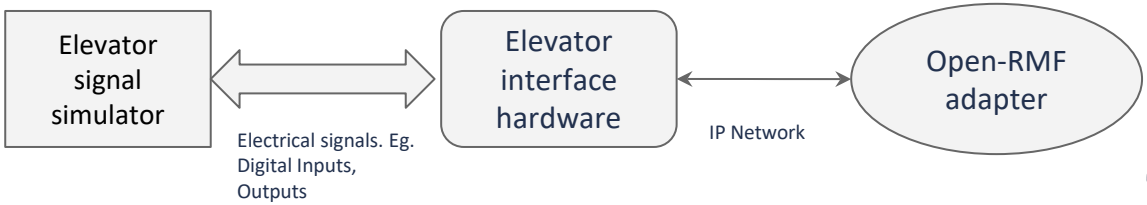
```
def door_command_mock(open):
    return True
```

Open-RMF

Testing with hardware

Strategies when **stateful hardware (eg. lifts)** are unavailable

- Connect to a simulated hardware
  - An elevator signal emulator for interface testing is available here: github.com/open-rmf/mock-lift-io



Open-RMF

3.

First run onsite

Connect to real hardware in the real environment, with ability to introspect

Open-RMF

Simulation

No hardware involved.
We can conduct repetitive testing using a fraction of the time for debugging.

Lab

Integrating RMF with real robots, putting the implemented fleet adapter to test.

Controlled environment

Running RMF at the actual site, discovering new limitations in the space, deployment network and building infrastructure we're dealing with.

Production

Co-exist with human staff along with scheduled building/robot maintenance, fire alarm evacuation exercises, etc.

Tune

Open-RMF

First run onsite

Treat this as a discovery phase

- Run a single robot through **all** traffic lanes and waypoints.
- Observe network connectivity (e.g. WiFi, LTE, etc.)
- Once the confidence is high, test with more robots.
- Test traffic conflicts, at least for all expected situations.
- Invite the client to test at critical milestones - they almost always are better at finding bugs!

Open-RMF

Our experiences

Limited and dynamic space

Building infrastructure

Limited charging points

IT constraints

Failed pickups

Lift travel

Periodic maintenance

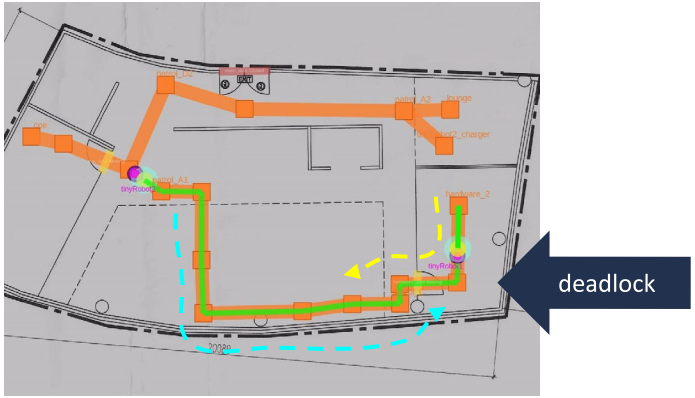
Connectivity

Open-RMF

Resolving constraints

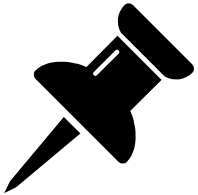
Space constraints 😟

- Long and narrow hallways
- Tiny rooms with limited space for even a single robot to move around
- Heavy foot traffic and dynamic environment
- Several mobile obstacles around



Open-RMF

Resolving constraints



~~Space constraints~~ 😐

Mutex groups 🙂 NEW FEATURE

To reduce potential negotiation deadlocks, users can now **assign a mutex to waypoints and lanes** to restrict occupancy of the area to only **one robot at a time**.

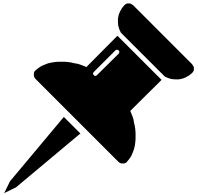
Without mutex groups

With mutex groups

New feature available in rmf_ros2 repository

Open-RMF

Resolving constraints



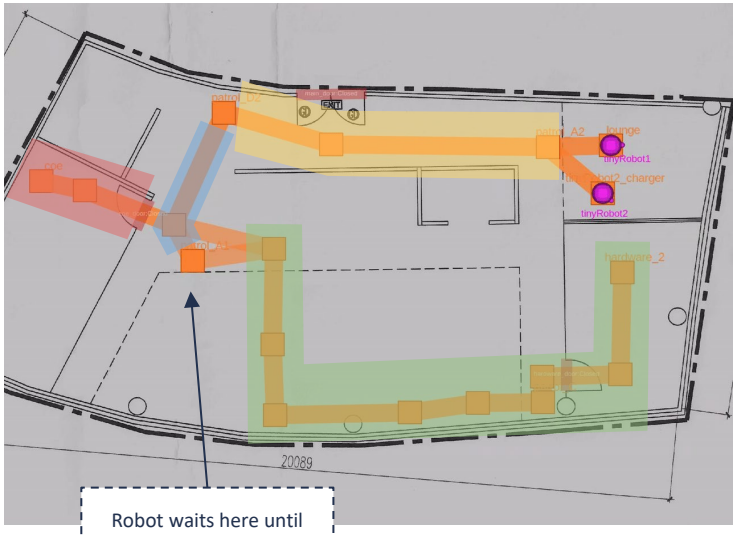
~~Space constraints~~ 😐

Mutex groups 🙂

In this example,

Mutex groups applied by adding a mutex property to lanes and waypoints.

This creates “zones” where only one robot can occupy.



Robot waits here until green mutex group is no longer occupied

Properties	
index	58
x (pixels)	2238.4710
y (pixels)	477.1230
x (m)	18.9458
y (m)	-4.0380
name	patrol_A2
mutex	door_corridor

Mutex applied on a **waypoint**

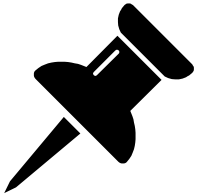
Properties	
edge_type	lane
start_idx	46
end_idx	65
start x (m)	18.7944
start y (m)	10.3724
end x (m)	20.8937
end y (m)	10.3084
length (m)	2.1002
bidirectional	true
demo_mock_floor_name	
demo_mock_lift_name	
mutex	window_corridor
speed_limit	0

Mutex applied on a **lane**

New feature available in rmf_ros2 repository

Open-RMF

Resolving constraints



~~Space constraints~~ 😐

Mutex groups 🙂

In this example,

Triangular roundabout added to manage movement between mutex groups.

This is achieved by using unidirectional lanes.

New feature available in rmf_ros2 repository

Open-RMF

Resolving constraints

Limited chargers ☹

This particular site had less chargers than robots, e.g. 1 charger for 2 robots.

This meant that the robots needed to share chargers, something that Open-RMF did not offer at the time.

Resolving constraints

~~Limited chargers~~ 😐

NEW FEATURE

Dynamic charging 🙂

Introducing the concept of a charging schedule:

Users get to **schedule each robot to a different charger during different periods** throughout the day.

Without dynamic charging

With dynamic charging

New feature available in rmf_ros2 repository

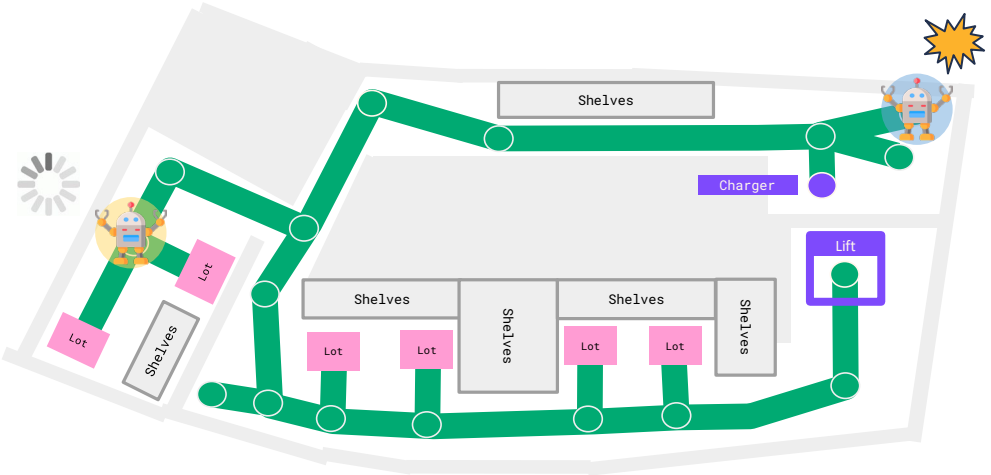
Open-RMF

Resolving constraints

Unable to remove robot from fleet during ops 😟

Robots need to be taken out of the fleet periodically for maintenance.

We would have to remove the robot from its RMF fleet to **prevent task allocation to an absent robot**.



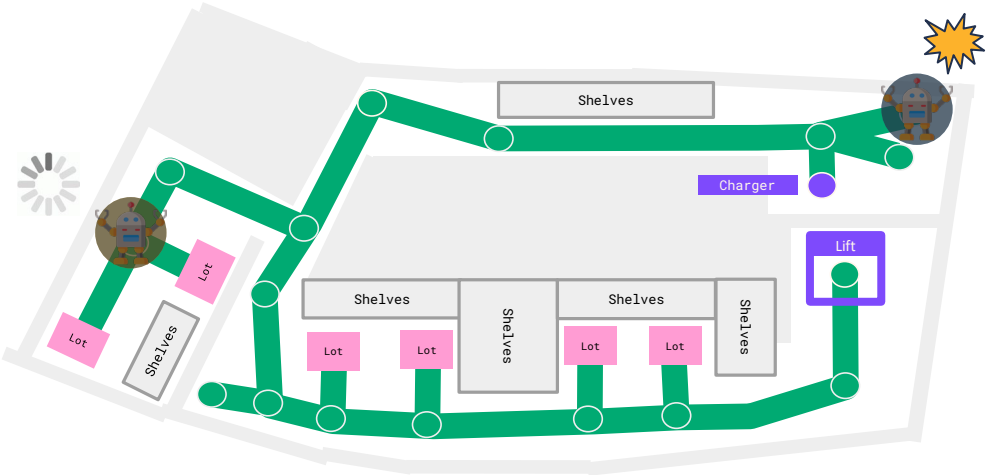
Open-RMF

Resolving constraints

Unable to remove robot from fleet during ops 😟

At the time, to remove one robot from the fleet adapter, we would need to bring down the entire fleet first.

If done during operational hours, this **disrupts the remaining robots in the fleet** and their tasks.



Open-RMF

Resolving constraints

~~Unable to remove robot from fleet during ops~~ 🙄

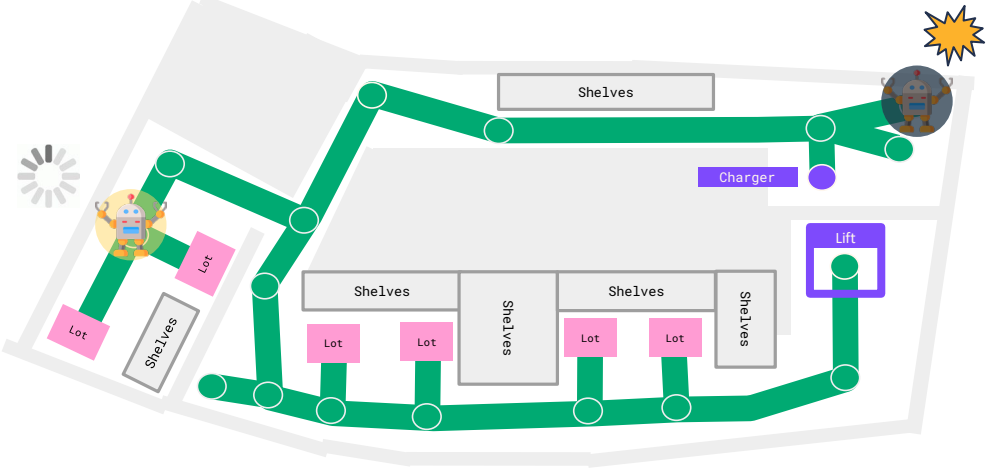
Robot commission 🙂

NEW FEATURE

We can now **decommission a robot without affecting other robots in the fleet**.

RMF will also **re-allocate any queued tasks** for this robot.

Users can re-commission the robot whenever ready.



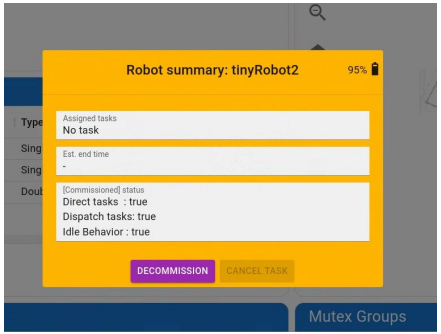
New feature available in rmf_ros2 repository

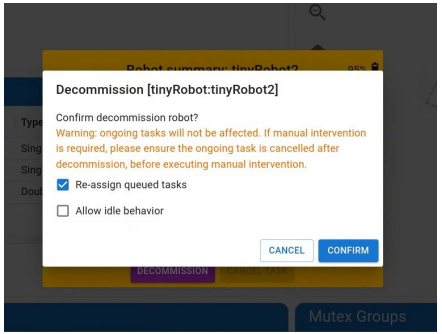
Resolving constraints

~~Unable to remove robot from fleet during ops~~ 🙄

Robot commission 🙂

Decommissioning can re-commissioning can be done from RMF Web.





Robots

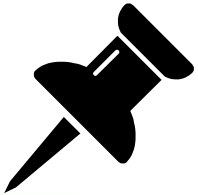
Name	Fleet	Est. Task Finish Ti...	Level	Battery	Last Updated	Status
tinyRobot1	tinyRobot	-	L1	95.00%	1/1/1970, 12:45:3...	IDLE
tinyRobot2	tinyRobot	-	L1	95.00%	1/1/1970, 12:45:3...	DECOMMISSIONED

1 row selected 1–2 of 2

New feature available in rmf_ros2 repository

Open-RMF

Resolving constraints



WiFi connectivity ☹

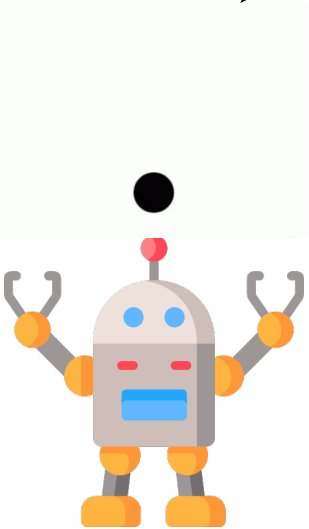
Robots disconnect from WiFi periodically while in motion.

Fleet adapter may not behave as desired without updates from robots.

WiFi connectivity ☺

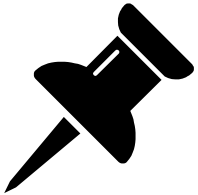
Test pings to server from robot, and restart WiFi if *ping loss > threshold*.

Add retry loops in fleet adapter to attempt to re-retrieve robot states and replan navigation paths.



Open-RMF

Resolving constraints



IT limitations ☹

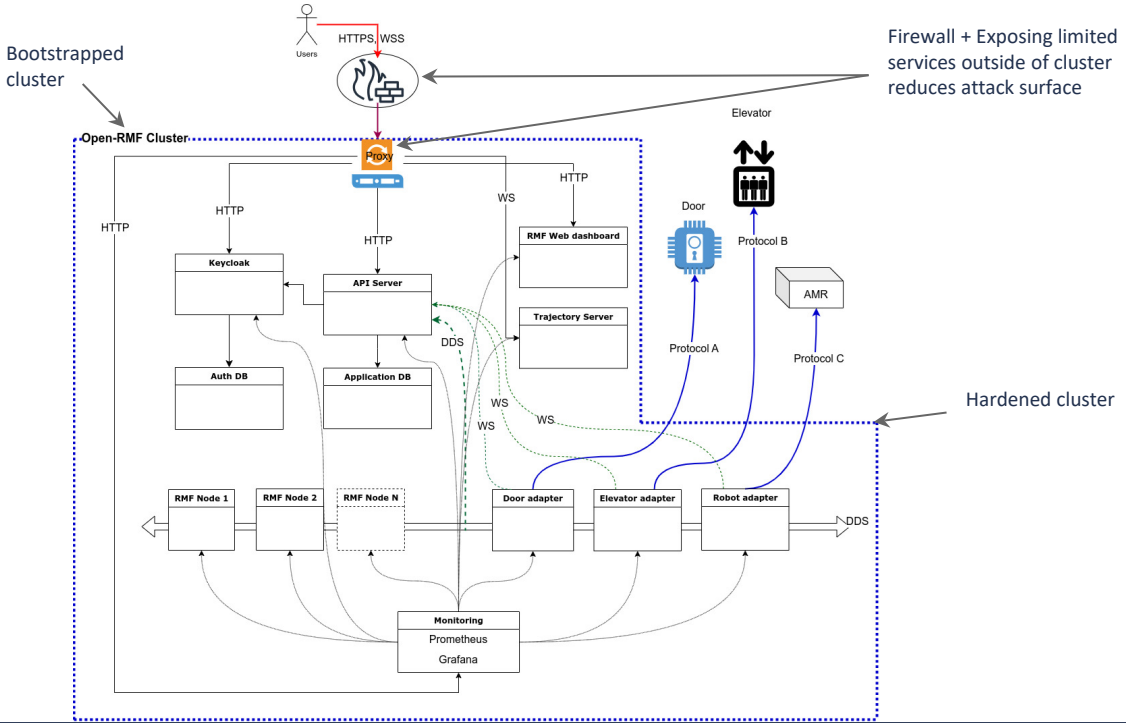
- Solutions must be hosted locally
- Only designated firewall ports can be opened.
- Data on server should not be “seen” by a malicious actor if gained access to host

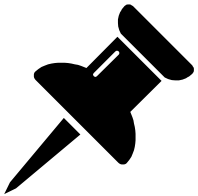
IT resolutions ☺

- Bootstrapped kubernetes cluster locally using k3s.
- Restricted external communications to run fully over port 443 (robot uses REST API), and ran DDS in-cluster only (Cyclone DDS worked well for our deployments).
- Cluster hardening to limit data exposure.

Open-RMF

Resolving constraints





Open-RMF

4.

Calling it in production

You have tested well, and are now ready to go live!

Open-RMF

Simulation

No hardware involved.
We can conduct repetitive testing using a fraction of the time for debugging.

Lab

Integrating RMF with real robots, putting the implemented fleet adapter to test.

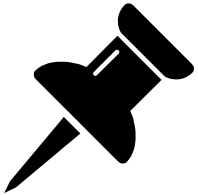
Controlled environment

Running RMF at the actual site, discovering new limitations in the space, deployment network and building infrastructure we're dealing with.

Production

Co-exist with human staff along with scheduled building/robot maintenance, fire alarm evacuation exercises, etc.

Certify Open-RMF



Calling it in production

Run tests to include special cases in your UAT

- Ask your customer to play with the deployment!
- For rare events like fire alarm triggers, ask the site to trigger it for testing with robots - observe if the robot performs the intended behaviors.
- For lift travel, test with human interference.

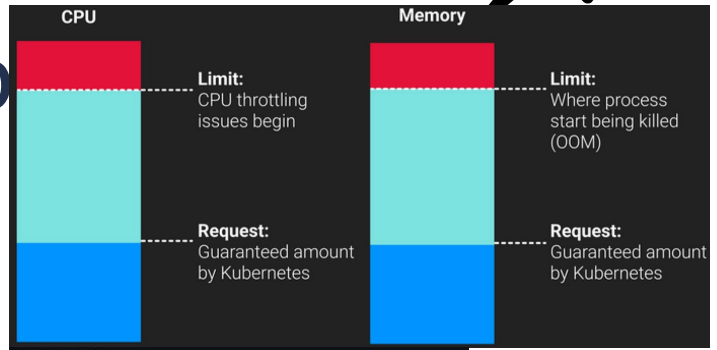
Open-RMF

Use Case	Test Case	Test Method	Expected Result	Pass / Fail
User Login - Authorized User	Validate authorized user able to login	1. Navigate to login screen 2a. Enter correct ID and correct password 2b. Enter correct ID and wrong password 2c. Enter unauthorised ID and correct password 2d. Enter wrong ID and wrong password 3. Click "Login"	Dashboard authentication correctly and allows validated users to login via their ID. 2a → User should be able to login to dashboard 2b → User should not be able to login to dashboard 2c → User should not be able to login to dashboard 2d → User should not be able to login to dashboard	
Workflow 1 - Ad hoc delivery	Validate ad hoc job is added to task list and robot executes task immediately	1. Click "+ NEW TASK" button 2. Select Task type "Delivery" 3. Select Pickup Location 4. Select Dropoff Location 5. Click "Submit Now"	- Task should be created and listed in task view - Robot executes task	
Deconfliction	Validate two Robot will be able to de-conflict when cross path at lobby	No intervention	2 Robots will autonomously take turns to proceed to deconflict itself without human intervention.	

Calling it in productio

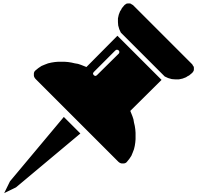
Tune resource limits and requests in charts

- E.g. A three-robot deployment across a few floors, is unlikely to require more than 4 CPU, 8 GB memory; spread allocation and limit SPoF.




```
containers:
  - name: rmf-building-map-server
    image: {{ .Values.builder_ns }}/rmf-site:{{ .Va
    imagePullPolicy: Always
    resources:
      requests:
        memory: "128Mi"
        cpu: "0.1"
      limits:
        memory: "256Mi"
        cpu: "0.25"
```

Open-RMF



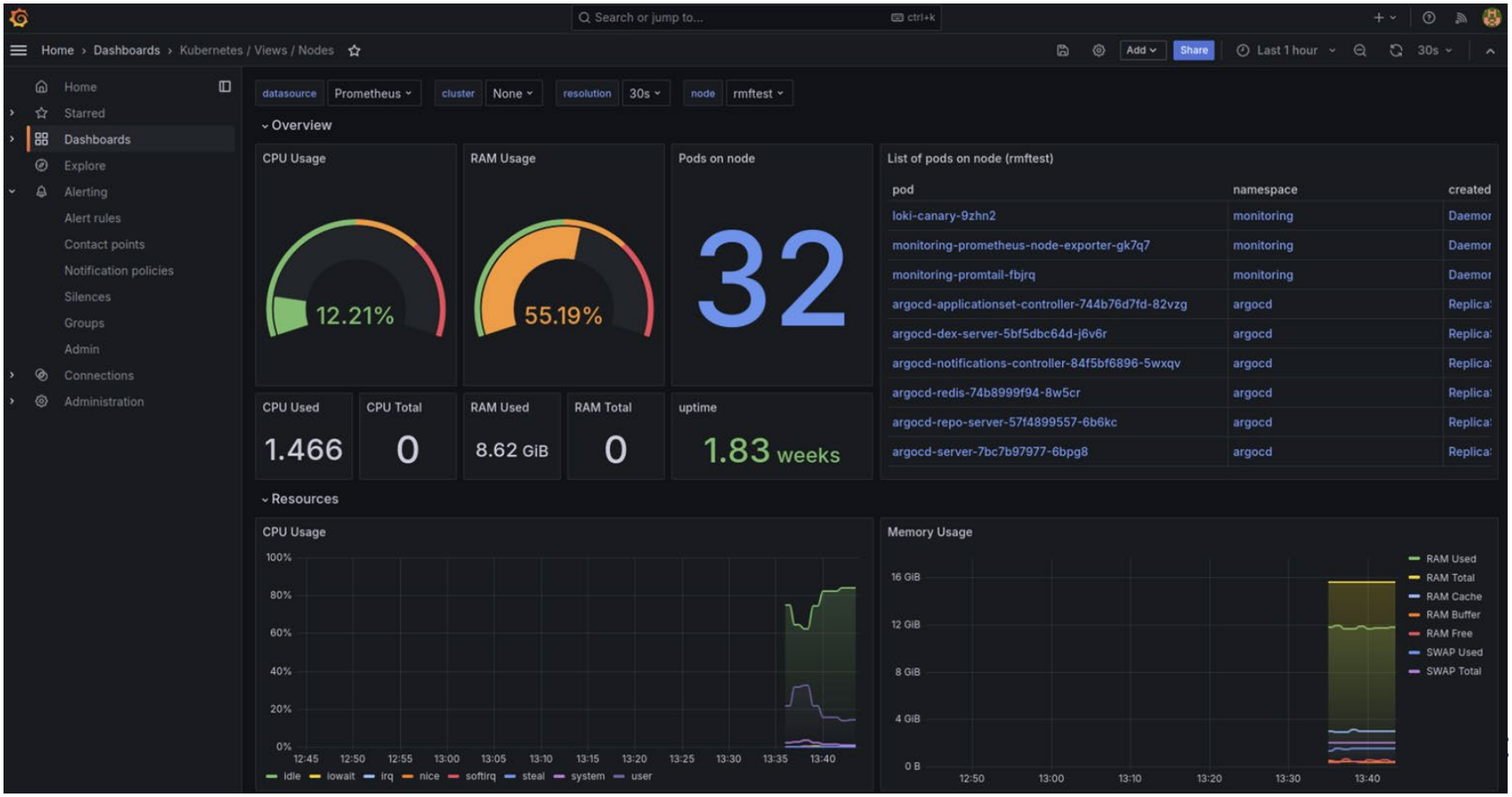
Calling it in production

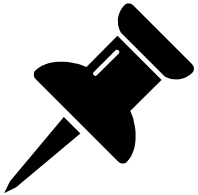
Pre-emptive alerting

- Running Open-RMF in test mode assumes somebody is watching all its moves, well not in production…
- Setup Grafana/Prometheus to send you alerts

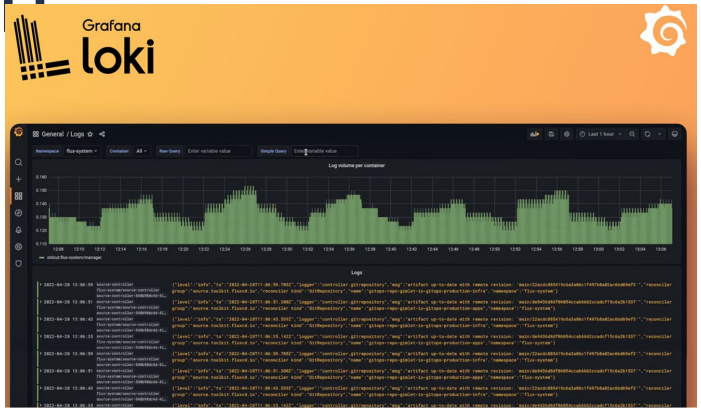
(yes, available with the deployment template!)

Open-RMF


Search or jump to... ctrl+k
Home › Dashboards › Kubernetes / Views / Nodes
Add
Share
Last 1 hour
30s
Home
Starred
Dashboards
Explore
Alerting
Alert rules
Contact points
Notification policies
Silences
Groups
Admin
Connections
Administration
datasource Prometheus
cluster None
resolution 30s
node rmftest
Overview
CPU Usage
12.21%
RAM Usage
55.19%
Pods on node
32
List of pods on node (rmftest)
pod
namespace
created
loki-canary-9zhn2
monitoring
Daemor
monitoring-prometheus-node-exporter-gk7q7
monitoring
Daemor
monitoring-promtail-fbjrq
monitoring
Daemor
argocd-applicationset-controller-744b76d7fd-82vzg
argocd
Replica
argocd-dex-server-5bf5dbc64d-j6v6r
argocd
Replica
argocd-notifications-controller-84f5bf6896-5wxqv
argocd
Replica
argocd-redis-74b8999f94-8w5cr
argocd
Replica
argocd-repo-server-57f4899557-6b6kc
argocd
Replica
argocd-server-7bc7b97977-6bpg8
argocd
Replica
CPU Used
1.466
CPU Total
0
RAM Used
8.62 GiB
RAM Total
0
uptime
1.83 weeks
Resources
CPU Usage
100%
80%
60%
40%
20%
0%
12:45
12:50
12:55
13:00
13:05
13:10
13:15
13:20
13:25
13:30
13:35
13:40
idle
iowait
irq
nice
softirq
steal
system
user
Memory Usage
16 GiB
12 GiB
8 GiB
4 GiB
0 B
12:50
13:00
13:10
13:20
13:30
13:40
RAM Used
RAM Total
RAM Cache
RAM Buffer
RAM Free
SWAP Used
SWAP Total



Calling it in production



Logging for future debugging

- The importance of logs can never be overstated!
- Setup Loki to log pod outputs and rotate periodically

(yes, available with the deployment template!)

Open-RMF

Thanks!

Special thanks to everyone in the Intrinsic Singapore team, who made a robust Open-RMF deployment possible!

rmf_deployment_template
Github repository

rmf_ros2
Github repository

Find out more about
Open-RMF

Open-RMF