Robotics Middleware Framework

Marco A. Gutiérrez

marco@openrobotics.org

open robotics

January 2021

The next frontier for ROS: multi-robot systems

The next frontier for ROS: multi-robot systems

- Robotics is a rapidly-developing field
  - delivery is a common application today
  - new applications will emerge
  - new companies will emerge
  - facilities will need to quickly adapt

- Certain process flow automations inherently require two or more robots to interact with each other
  - best robots for each sub-task may come from different vendors
  - **multi-vendor integration** must be possible!

Multi-fleet Integration is Challenging

Lack of Interoperability

- Lack of communication and integration between robots, medical devices, building infrastructure and health IT systems

Infrastructure Constraints

- Need to interface with lifts and doors
- Dedicated routes and lifts for robot

Lack of Realistic Test Environment

- Challenging and expensive to test effectiveness of large scale deployment of robotic solutions

Dynamic Environments

- Dynamic human traffic and crowds
- Direct contact with high volume of untrained personnel and visitors

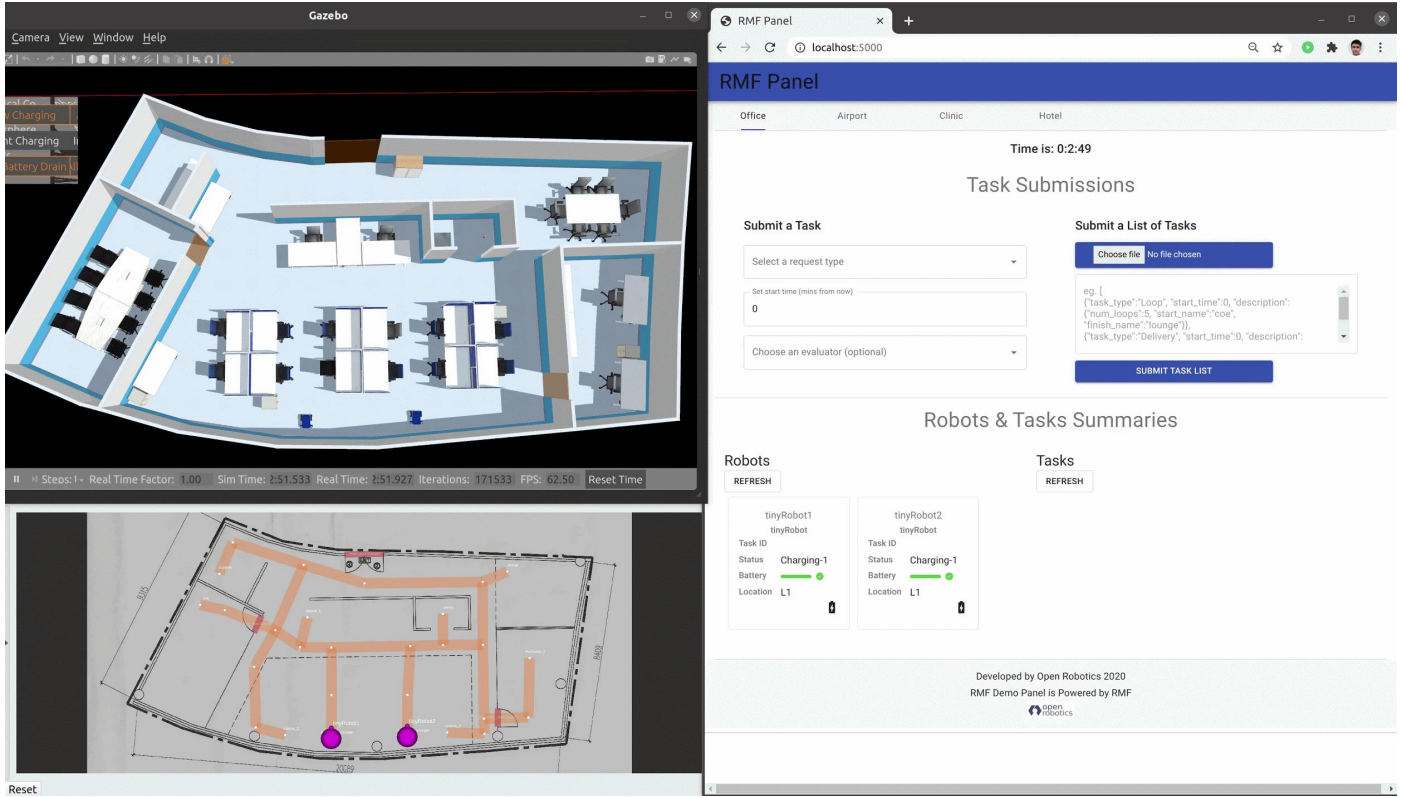
Cybersecurity Concerns

- Increased reliance on network for data transmission

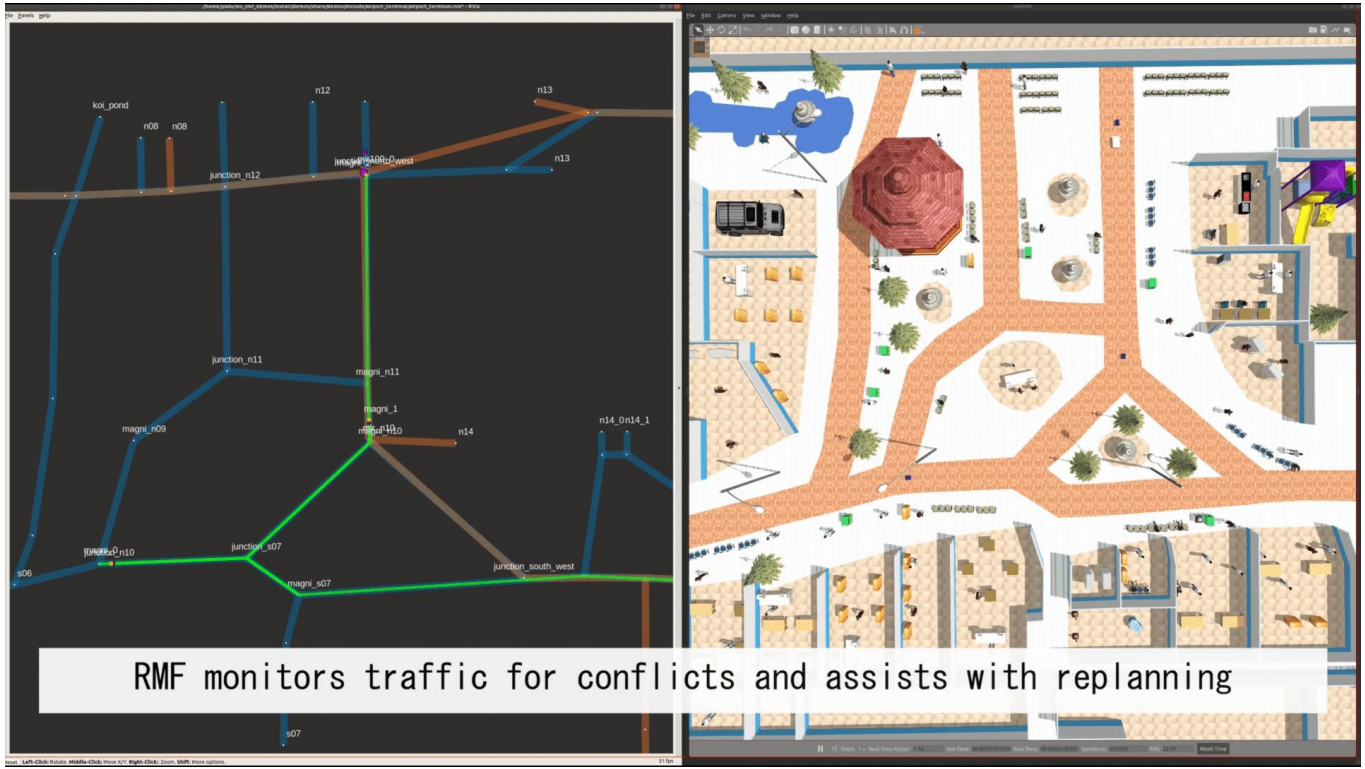
open robotics

What can RMF do?

Task planning and allocation

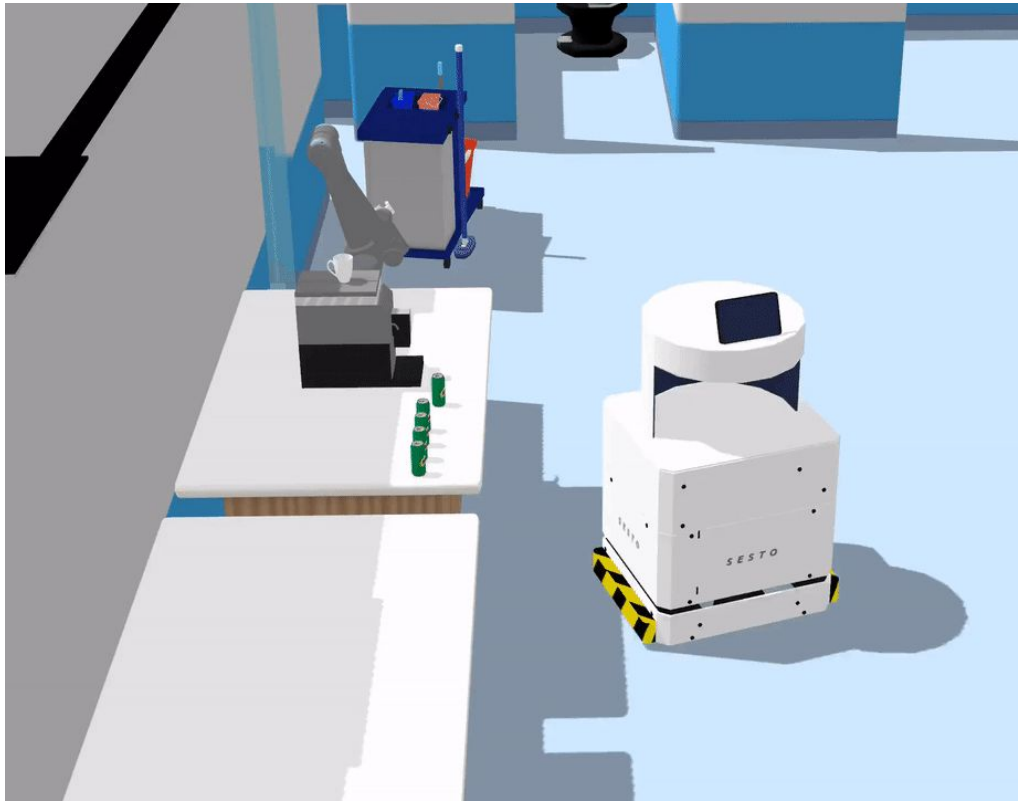


Manage Fleet Traffic



Door/Lift integration

Interact with workcells



open robotics

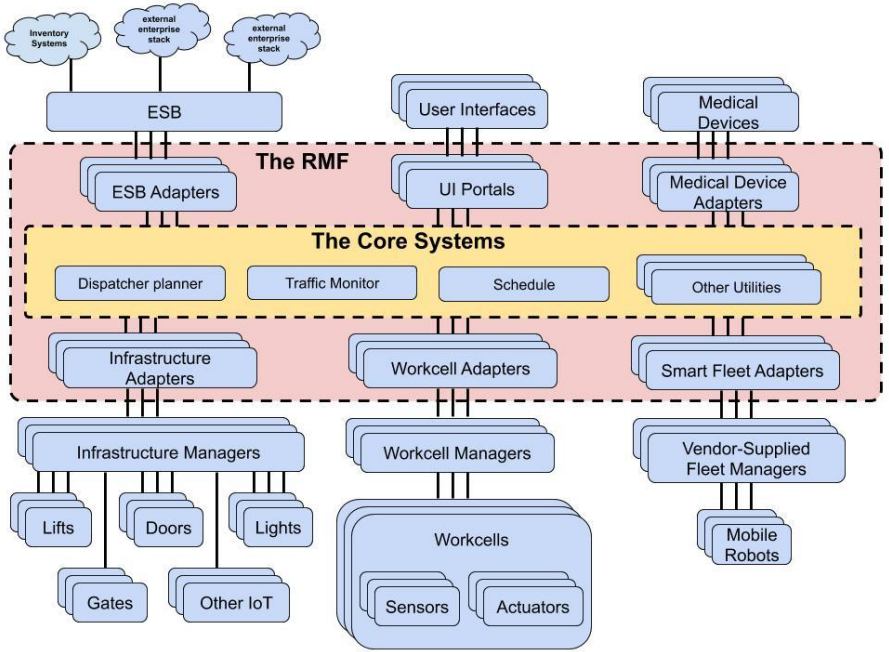
How does RMF work?

What is RMF?

Robotics Middleware Framework is a collection of libraries and tools that facilitate interoperability among

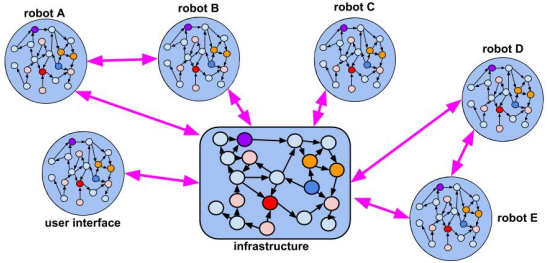
- heterogeneous robot fleets
- building infrastructure systems (door, lifts, etc)
- automation systems (dispensers, collectors, etc)

It adds intelligence to the system through resource allocation and by preventing conflicts over shared resources.

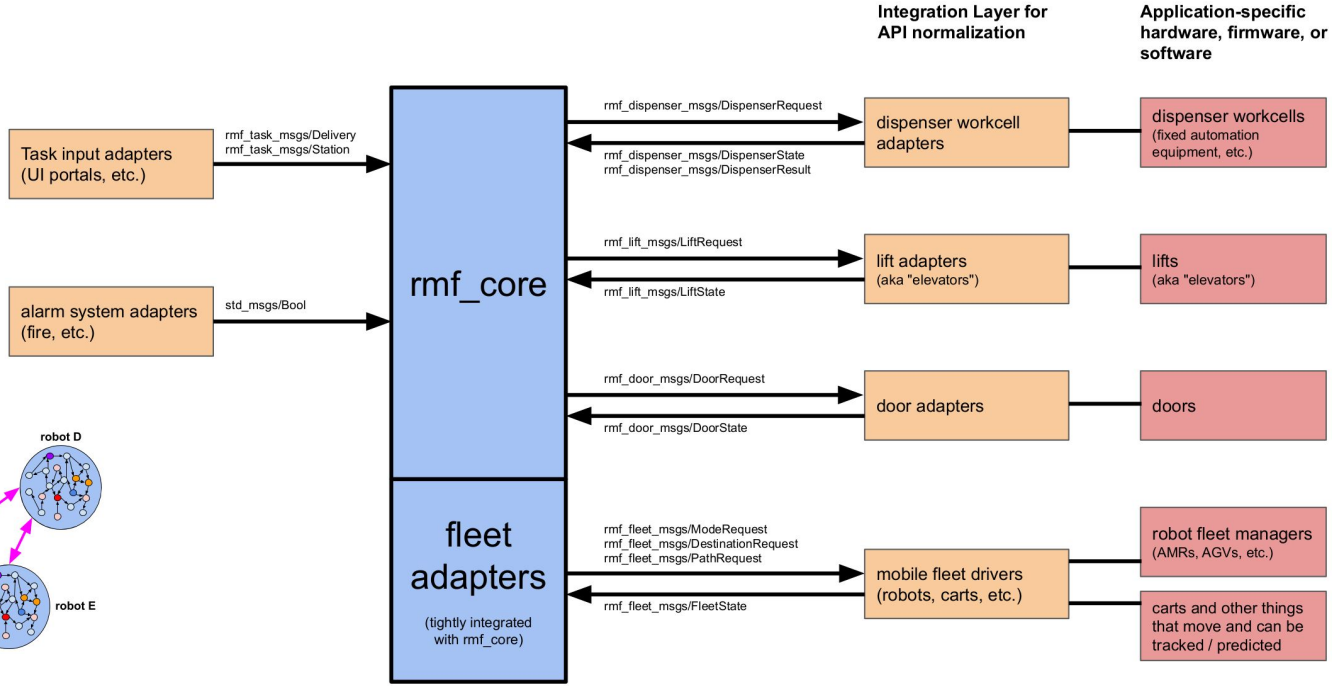


http://github.com/osrf/rmf_core

RMF Simplifies & Standardizes Messaging



Systems of Systems Synthesizer (SOSS)

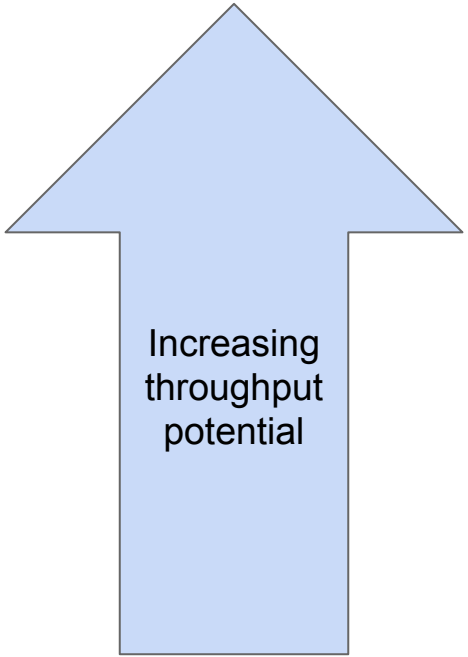


Messaging architecture in RMF

open robotics

http://github.com/osrf/traffic-editor

How much control is needed? It's complicated.

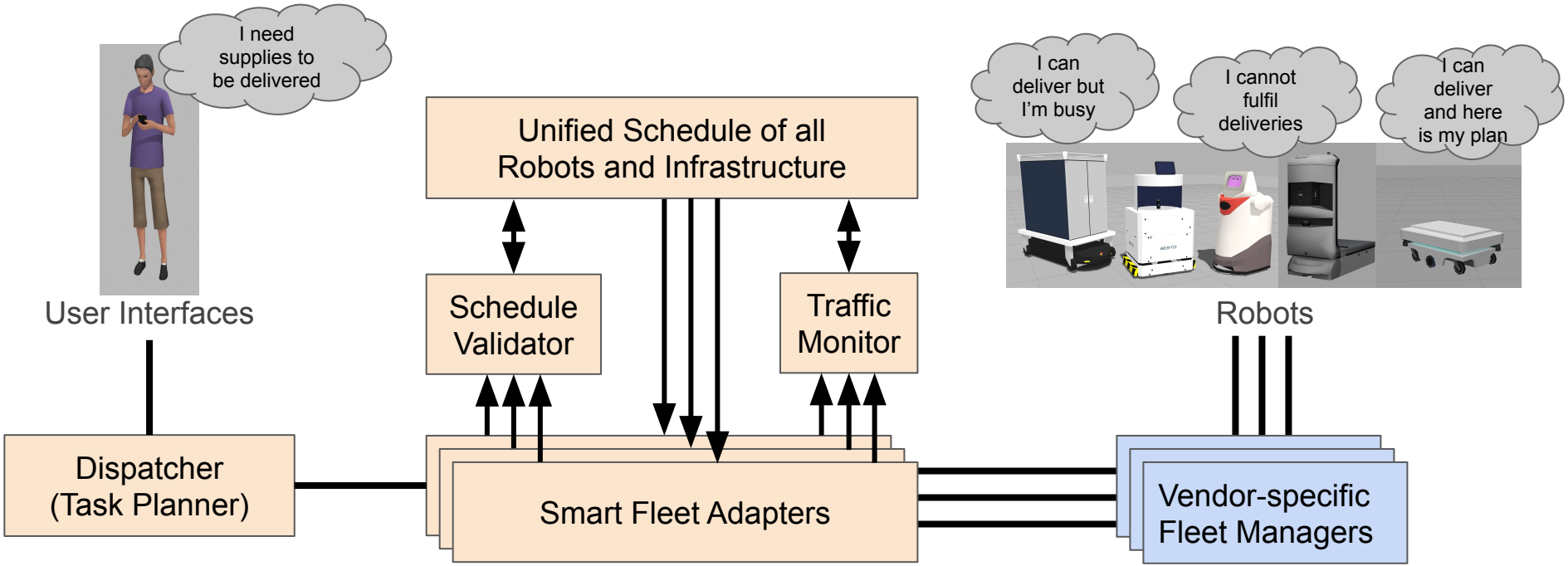


Fleet Manager API features	Potential benefits
"paths": robot waypoint control	Reduce stoppages, increase interaction efficiency. Deal gracefully with the unexpected.
"traffic light": robot pause, resume	Spatio-temporal separation of robots can often be achieved.
"read-only": robot locations, destinations	Unified dashboard for operators. Data can allow other robots to avoid conflicts with this fleet.
no fleet API ☹	None. Fleets cannot coordinate with each other at all to avoid or resolve conflicts.

open robotics

http://github.com/osrf/rmf_core

RMF abstracts tasks for improved flexibility

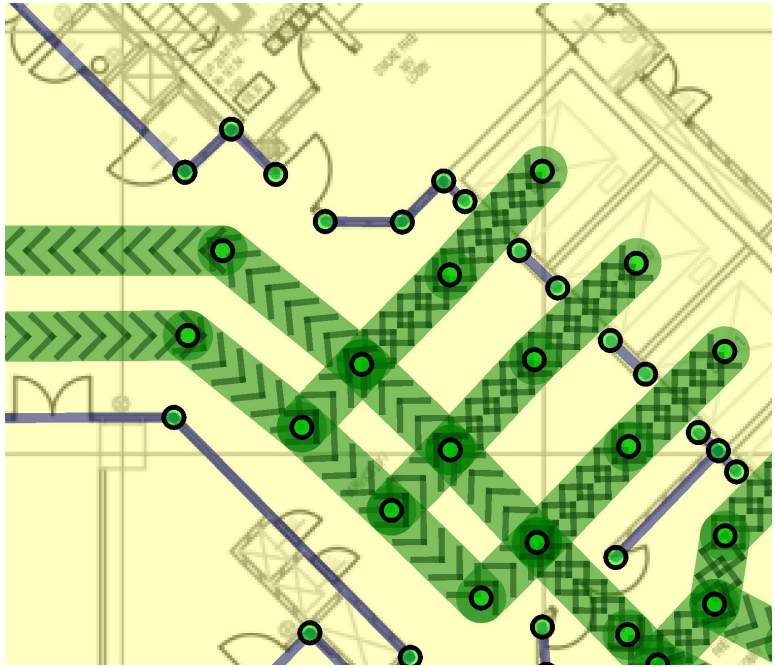


open robotics

http://github.com/osrf/rmf_core

RMF introduces traffic conventions

- fleets follow a common traffic lane layout
  - minimize the fleet interactions
  - simplify adding new robots to traffic flow
- fleets follow traffic management commands
- fleets share mechanical infrastructure
  - access to elevators/doors is provided by a higher-level controller
- fleets implement behaviors for emergencies
  - code blue, fire alarm, etc.
  - typically "get out of the way and stop moving"



RMF predicts and resolves conflicts

Predicted plan for a robot may change given environmental variability

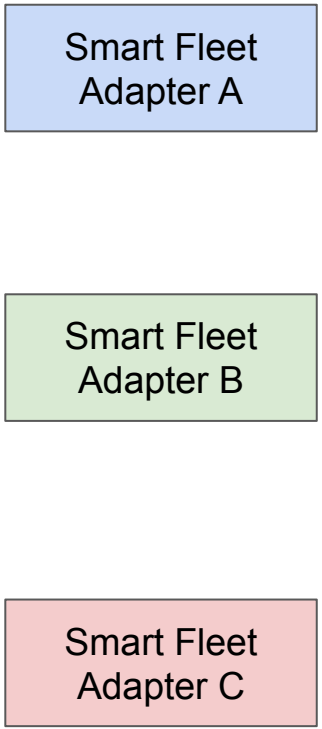
- Human traffic
- Unknown obstacles
- Busy/faulty lifts or doors
- Change in original intention
- Emergency stop or alarms

RMF continuously monitors changes in the Unified Schedule and appropriately reports conflicts to Smart Fleet Adapters for resolution.

open robotics

http://github.com/osrf/rmf_core

Traffic Negotiation



Assumptions:

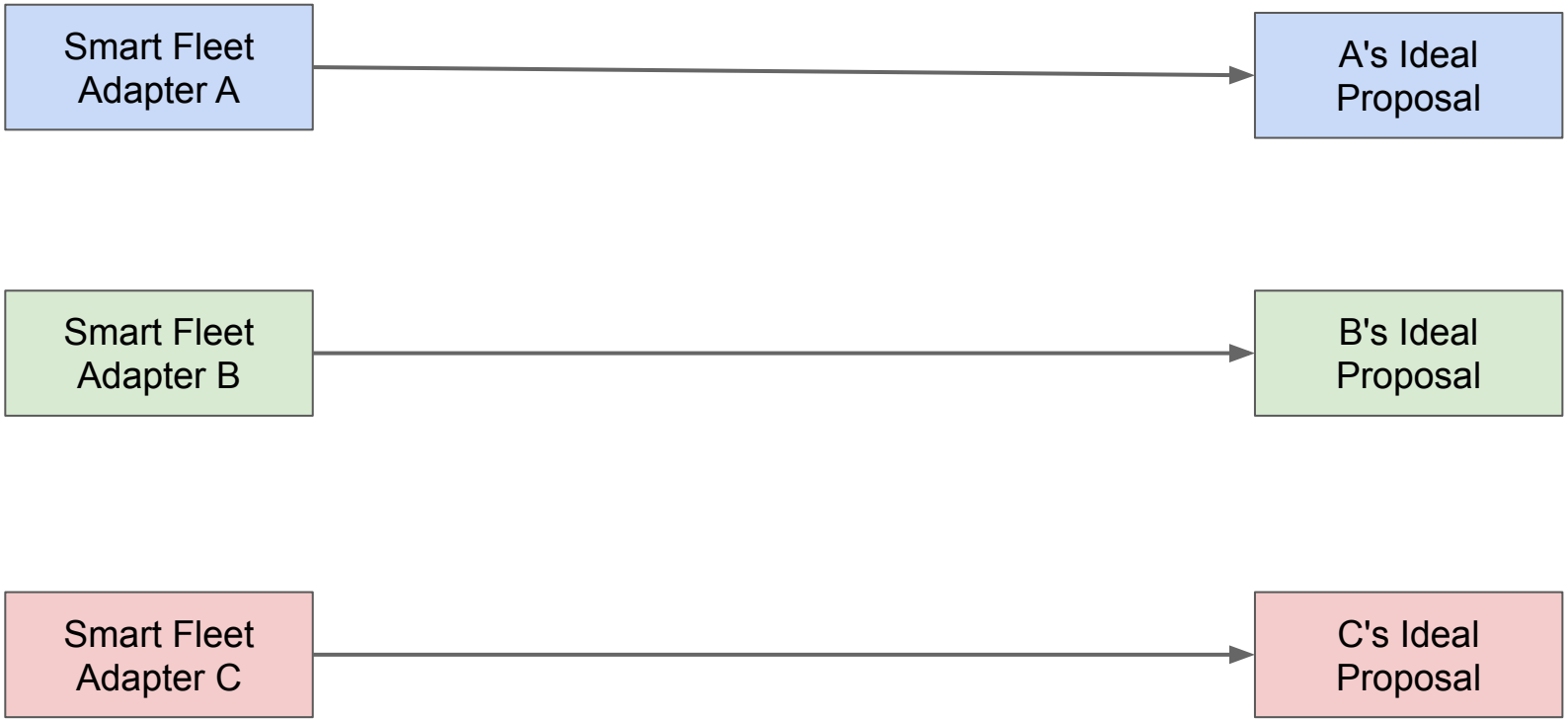
- Each fleet does NOT know what the other is capable of
- Each fleet can communicate a plan that is feasible for itself
- Each fleet can see the other's plans and attempt to plan around it

open robotics

http://github.com/osrf/rmf_core

Traffic Negotiation

Each fleet proposes the itinerary they would like to follow

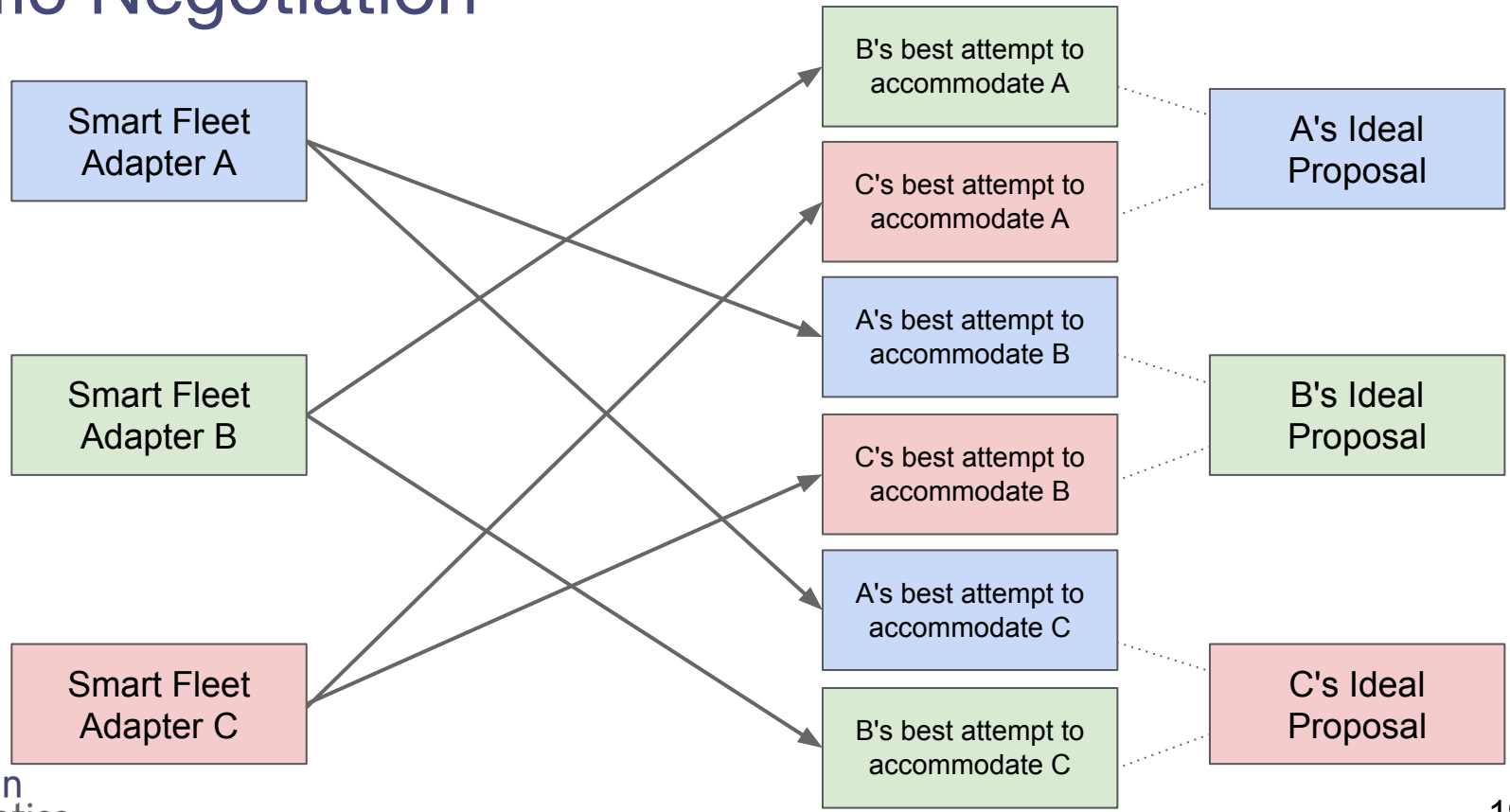


open robotics

http://github.com/osrf/rmf_core

Traffic Negotiation

Each fleet responds to the ideal itineraries of the others with an itinerary that is feasible for itself while accommodating the other

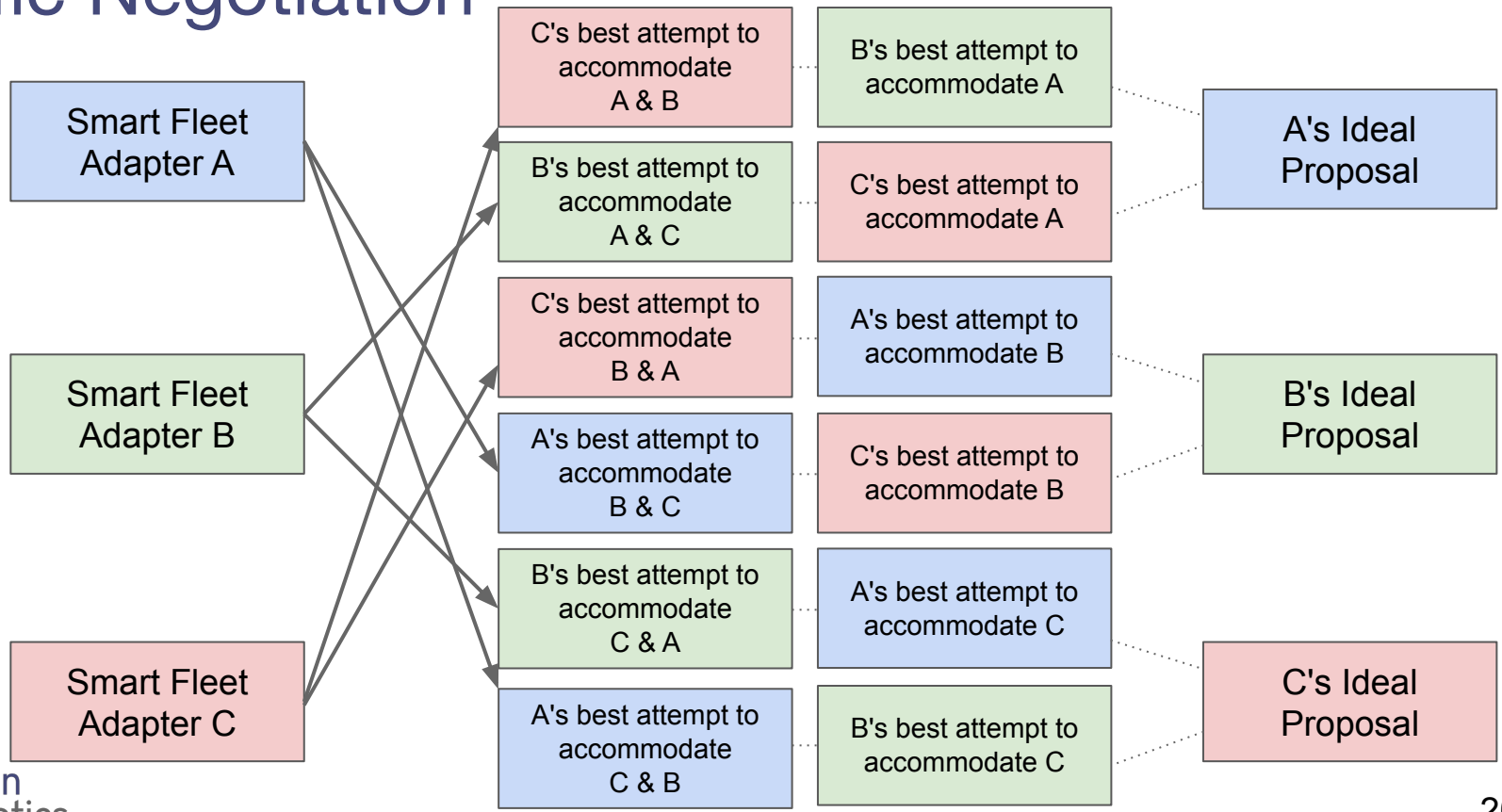


open robotics

http://github.com/osrf/rmf_core

Traffic Negotiation

Each fleet responds to each combination of the others' proposed itineraries with an itinerary that would feasible for itself



open robotics

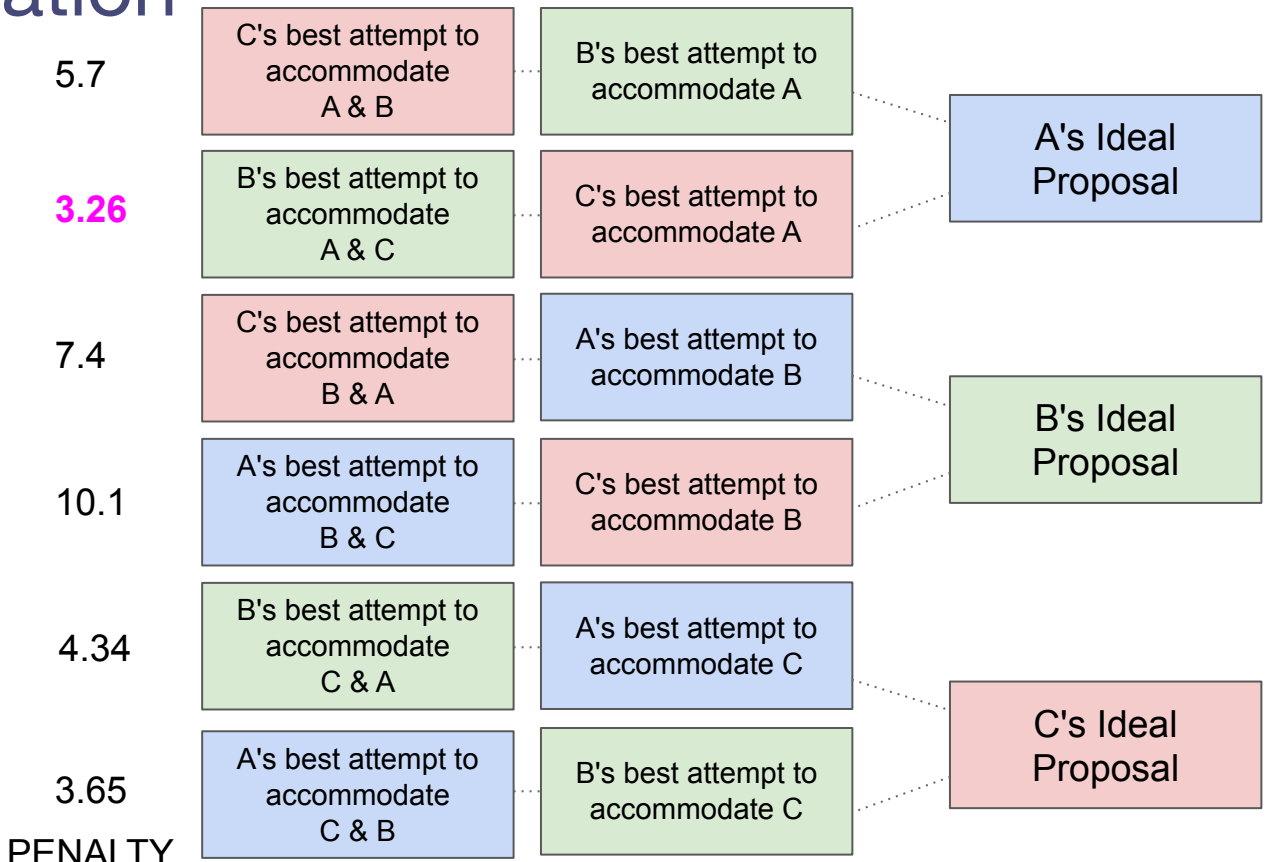
http://github.com/osrf/rmf_core

Traffic Negotiation

A third-party judge measures the penalty of each set of proposals.

The plan with the lowest penalty is chosen.

The penalty may be measured by the sum of the delays in completing all of the tasks. The sum may be weighted by the importance of each task.



open robotics

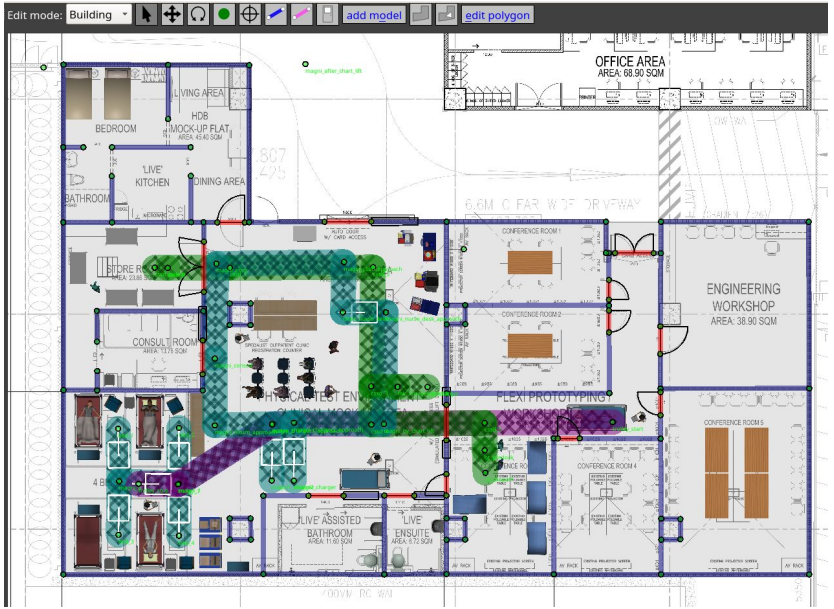
RMF Toolbox

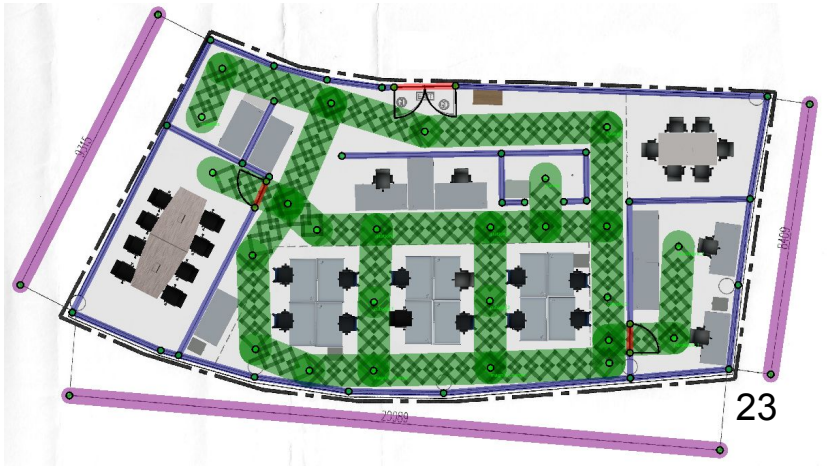
traffic editor | simulation tools | rmf core | UI

http://github.com/osrf/traffic-editor

Traffic Editor

- annotate floor plans
  - walls
  - doors
  - lifts
- place simulation models
  - static model placements
  - dynamic models (humans, etc.)
- import/export traffic lanes to vendor-specific Fleet Adapters

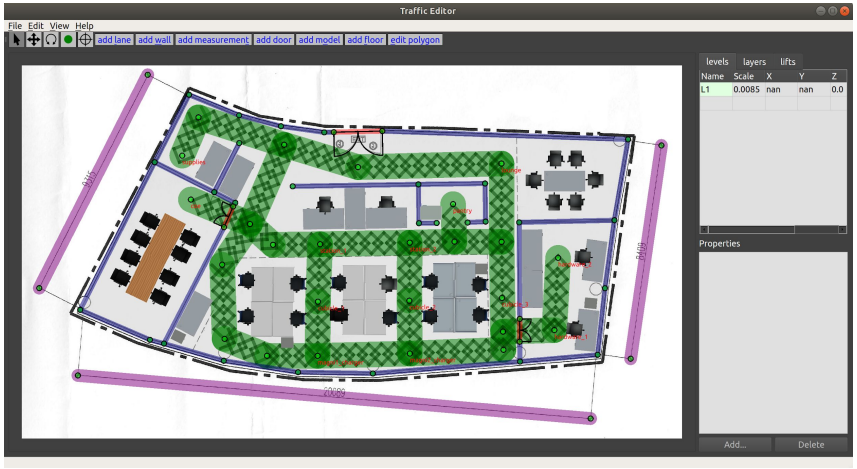




open robotics

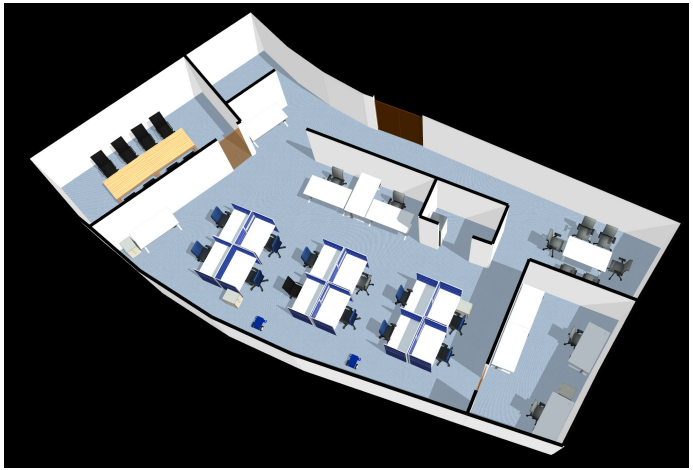
http://github.com/osrf/traffic-editor

Building map tools



Annotated map in ***traffic_editor***

building_map_tools



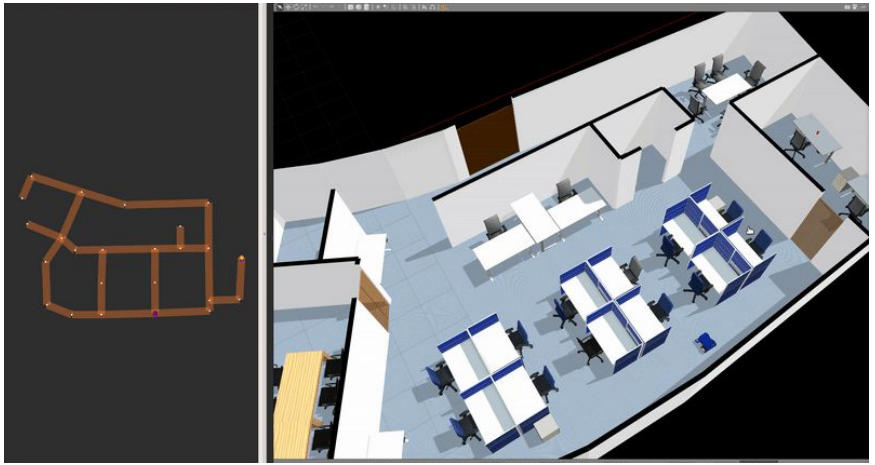
Physics-based simulation world with 3d assets, robots, doors, plugins

open robotics

http://github.com/osrf/rmf_demos

Testing in simulation is extremely important

- Time saving
- Fine tune algorithms
- Testing
  - Extended operation duration
  - Scalability
  - Debug edge cases
  - Vendor integration
- Using ROS and Gazebo, the code running in simulation is identical to that running in actual hardware!



Loop Request Scenario: Robots loop between waypoints while resolving conflicts in their paths and interacting with doorways

This is a physics-based simulation running the actual rmf_core software!

open robotics

https://app.ignitionrobotics.org/fuel

Many open-source simulation assets are available

(This is only a tiny sample)

open robotics

http://github.com/osrf/rmf_core

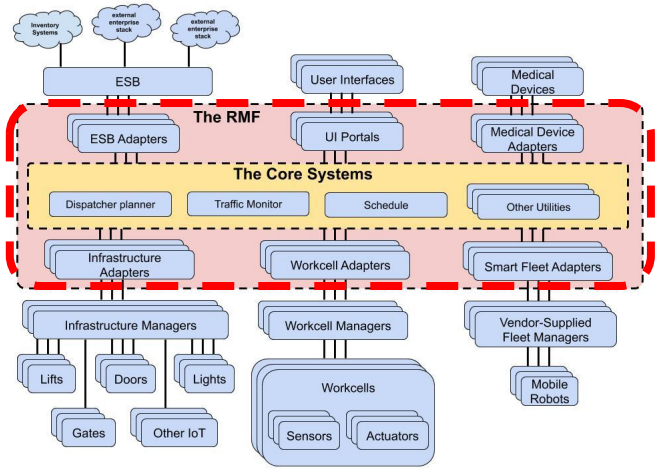
RMF Core is the brain of RMF

A collection of libraries and utilities for assisting vendors integrate with RMF

- Pure C++ libraries for
  - Trajectory interpolation
  - Path planning
  - Schedule database management
  - Conflict detection and resolution

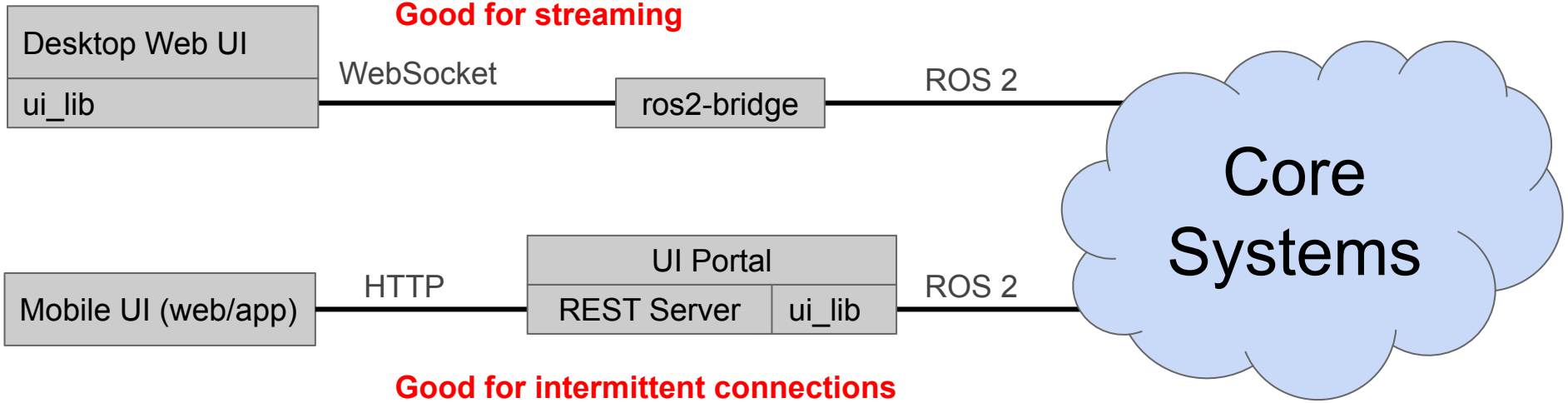
- ROS2 wrappers of C++ libraries

- Templates and examples of
  - Smart Fleet Adapters for fleets with various levels of control
  - Lift/Door adapters
  - Abstract task configurations



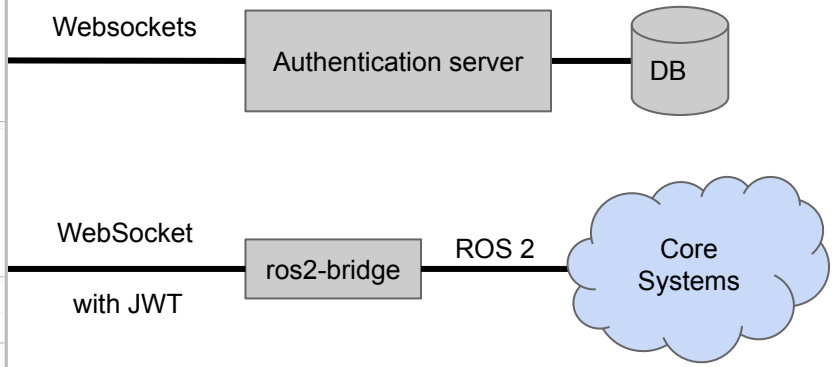
open robotics

UI Signal Paths



https://github.com/osrf/romi-dashboard

Operations Dashboard



- Robot fleet, door and lift state monitoring
- Schedule and trajectory visualization
- High-level commands to controllable assets, eg. robots, doors, dispensers, etc.

open robotics

Demo time!

Security Challenges

Security Challenges

1. 3rd parties hardware and software
2. Compromised recovery (CRLs support)
3. Intrusion detection tools
4. Security management dashboard
5. Level of access for humans
6. Network security (cloud)
7. Deployment security testing and certification

Where to get Started

- RMF Demos:
  - https://github.com/osrf/rmf_demos
- Office secured demo:
  - https://github.com/osrf/rmf_demos/blob/master/docs/secure_office_world.md
- RMF ros2 multi robot book:
  - https://osrf.github.io/ros2multirobotbook
- Free fleet:
  - https://github.com/osrf/free_fleet
  - https://osrf.github.io/ros2multirobotbook/integration_free-fleet.html
- RMF Core:
  - https://github.com/osrf/rmf_core
- Traffic Editor:
  - https://github.com/osrf/rmf_core

Thank you!