

# Open-RMFの仕様明確化

25<sup>th</sup>/Sep/2024

グエン ジュイヒン

**Panasonic**

Panasonic Asia Pacific Co. Ltd

# RMFの詳細

- RMFはROS2で構成されており、誰でも使用可能

RMFのソース

<https://github.com/open-rmf>

RMFのデモ環境

[https://github.com/open-rmf/rmf\\_demos](https://github.com/open-rmf/rmf_demos)

評価環境の作成

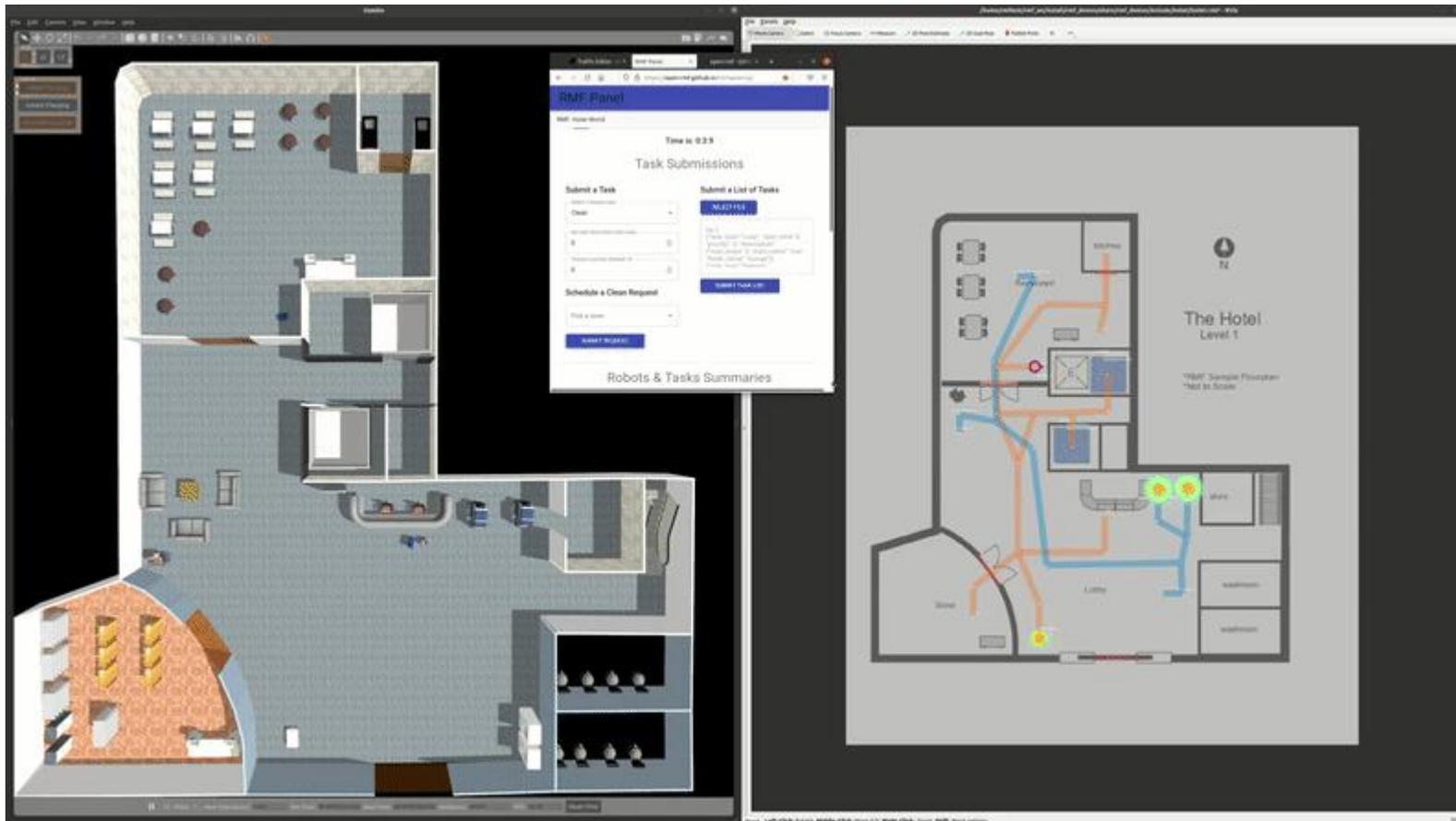
<https://osrf.github.io/ros2multirobotbook/traffic-editor.html>

RMFのドキュメント

<https://osrf.github.io/ros2multirobotbook>

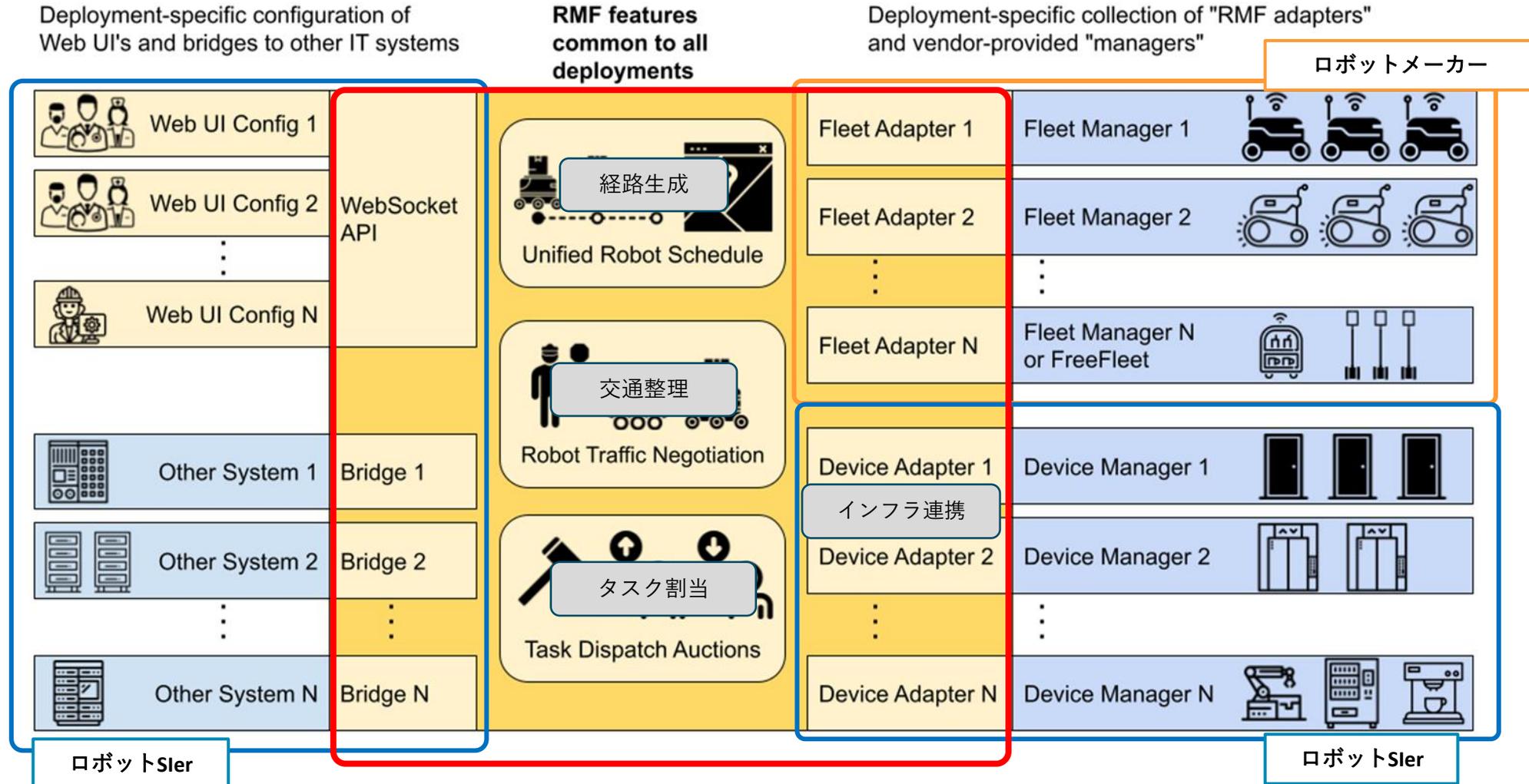
- 実際のロボットの動き  
(Gazebo)

- RMFで制御されているロボットの様子  
(Rviz)



# RMFの機能概要

ロボットを交通整理したり、インフラとの連携を実現する機能を備えているシステムに必要なUI、ロボット、インフラ、連携設備との接続部分はSIerによる開発が必要



# FMSに求められる機能一例とRMFの通常機能との比較

	機能	有/無	概要
1	現在位置から目的地までの経路生成	✓	現在位置から目的地までの経路を生成しロボットをその経路上を走行させる
2	特定経路における速度・向き制限制御	✓	特定の経路のみの特定の速度や向きを設定する
3	複数経路からの状況に応じた経路選択	✓	複数経路設定した場合、状況に応じて適切な経路を選定する
4	各ロボットの交通整理	✓	複数のロボットの進路が交差した場合優先順位付けを行う
5	自動ドアとの連携	✓	自動ドアとの連携を実現
6	エレベーターとの連携	✓	エレベーターとの連携を実現（単一リフト運用（Car call）のみ）
7	充電位置における自動充電シーケンス指示の発令	✓	充電器前における特別シーケンスの発令
8	目的地位置での停止角度指定	✗	目的位置での停止角度制御
9	シュミレーション環境簡易作成ツール	✓	評価するためのシュミレーション環境の作成
10	タスクのロボットへの割り当て	✓	特定のロボットに対するタスク割り当て
11	各ロボットの地図座標の統合	✓	各ロボットの座標系のGapを吸収
12	外部指令による全ロボットの退避位置への移動	✓	火災報知機などが鳴った際などにロボットを指定位置に退避
13	バッテリー残量が一定以下の場合、充電器に自動で戻る	✓	バッテリーが一定以下になったら強制的に充電器に帰還
14	エンドユーザー向けUI	△	エンドユーザーが操作したり、データを閲覧するUI
15	運用ログの保存	△	運用ログの保存

# FMSに求められる機能一例とRMFの通常機能との比較

	機能	有/無	概要
1	現在位置から目的地までの経路生成	✓	現在位置から目的地までの経路を生成しロボットをその経路上を走行させる
2	特定経路における速度制限制御	✓	特定の経路のみの特定の速度を設定する
3	複数経路からの状況に応じた経路選択	✓	複数経路設定した場合、状況に応じて適切な経路を選定する
4	各ロボットの交通整理	✓	複数ロボットの進路が交差した場合優先順位付けを行う
5	自動ドアとの連携	✓	自動ドアとの連携を実現
6	エレベーターとの連携	✓	エレベーターとの連携を実現（単一リフト運用（Car call）のみ）
7	充電位置における自動充電シーケンス指示の発令	✓	充電器前における特別シーケンスの発令
8	目的地位置での停止角度指定	✗	目的位置での停止角度制御
9	シュミレーション環境簡易作成ツール	✓	評価するためのシュミレーション環境の作成
10	タスクのロボットへの割り当て	✓	特定のロボットに対するタスク割り当て
11	各ロボットの地図座標の統合	✓	各ロボットの座標系のGapを吸収
12	外部指令による全ロボットの退避位置への移動	✓	火災報知機などが鳴った際などにロボットを指定位置に退避
13	バッテリー残量が一定以下の場合、充電器に自動で戻る	✓	バッテリーが一定以下になったら強制的に充電器に帰還
14	エンドユーザー向けUI	△	エンドユーザーが操作したり、データを閲覧するUI
15	運用ログの保存	△	運用ログの保存

ROSCON2023の発表を参考にしてください

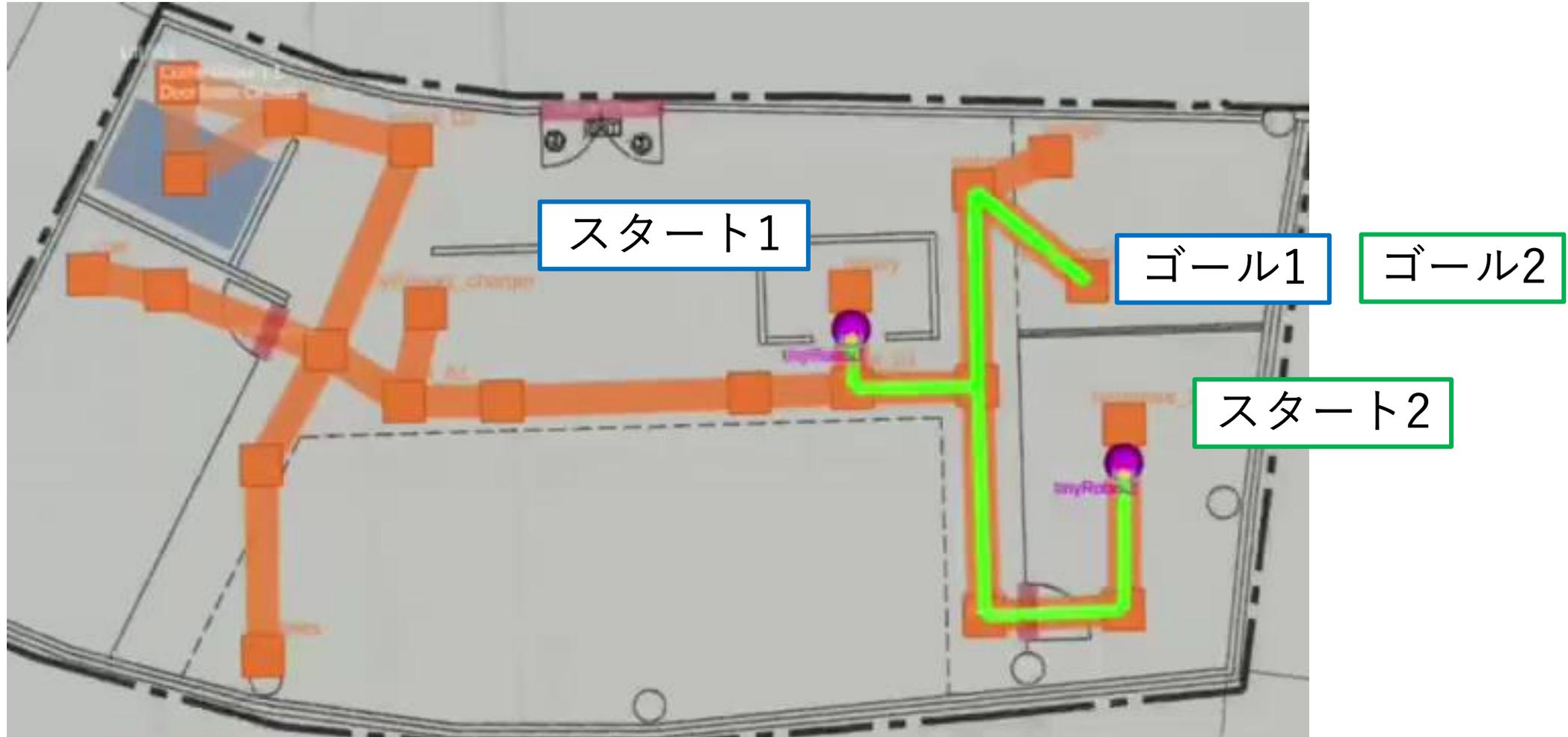
## (補足) 経路生成における問題点

実際の現場でロボットの走行位置・経路を微調整する際、pixel単位でしか調整不能  
RMFではロボットに対して走行禁止領域は設定不能（ロボット側で設定が必要）



## (補足) 交通整理における問題点

設定された経路にすれ違いを実施できる“解”が存在する事が前提  
“解”が存在しない場合はデッドロックが発生します



注意事項：同じタイミングで同じ目的地にはいけません（issueで報告済み）

# ロボットへのタスク割当

RMFにおけるロボットへのタスク割当は以下の2方法のみ

- ロボットのTypeを指定

該当のTypeに登録されている**“全”**ロボットから、RMFがある条件（たぶん、距離で判定）で自動的にタスク割り当て

- ロボットのIDを指定

**“特定のID”**のロボットに決め打ちでタスクを割り当て

Type :

Delivery

Cleaning

Concierge



ID :

Pink

Green

Blue

A

B

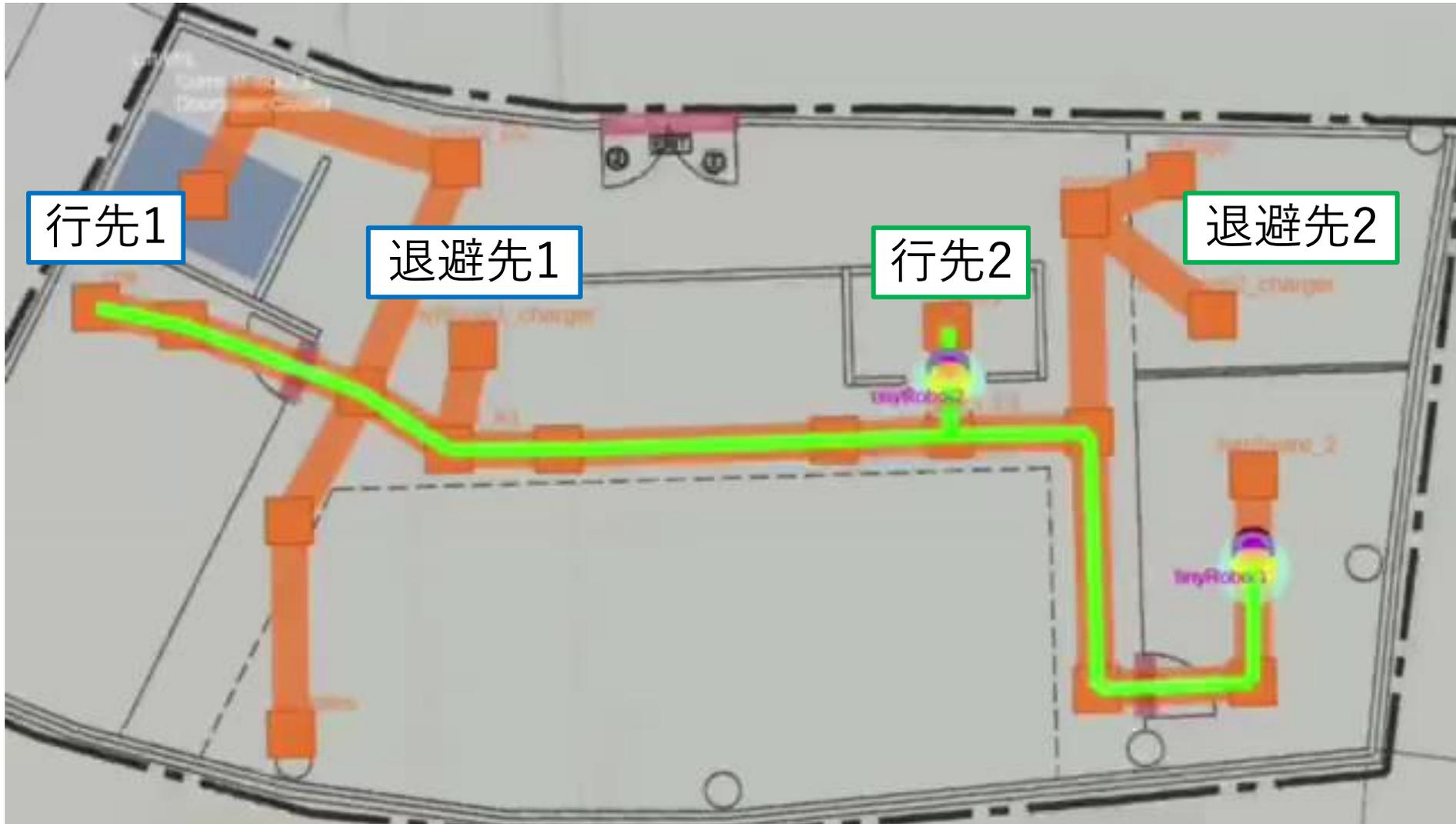
Black

White

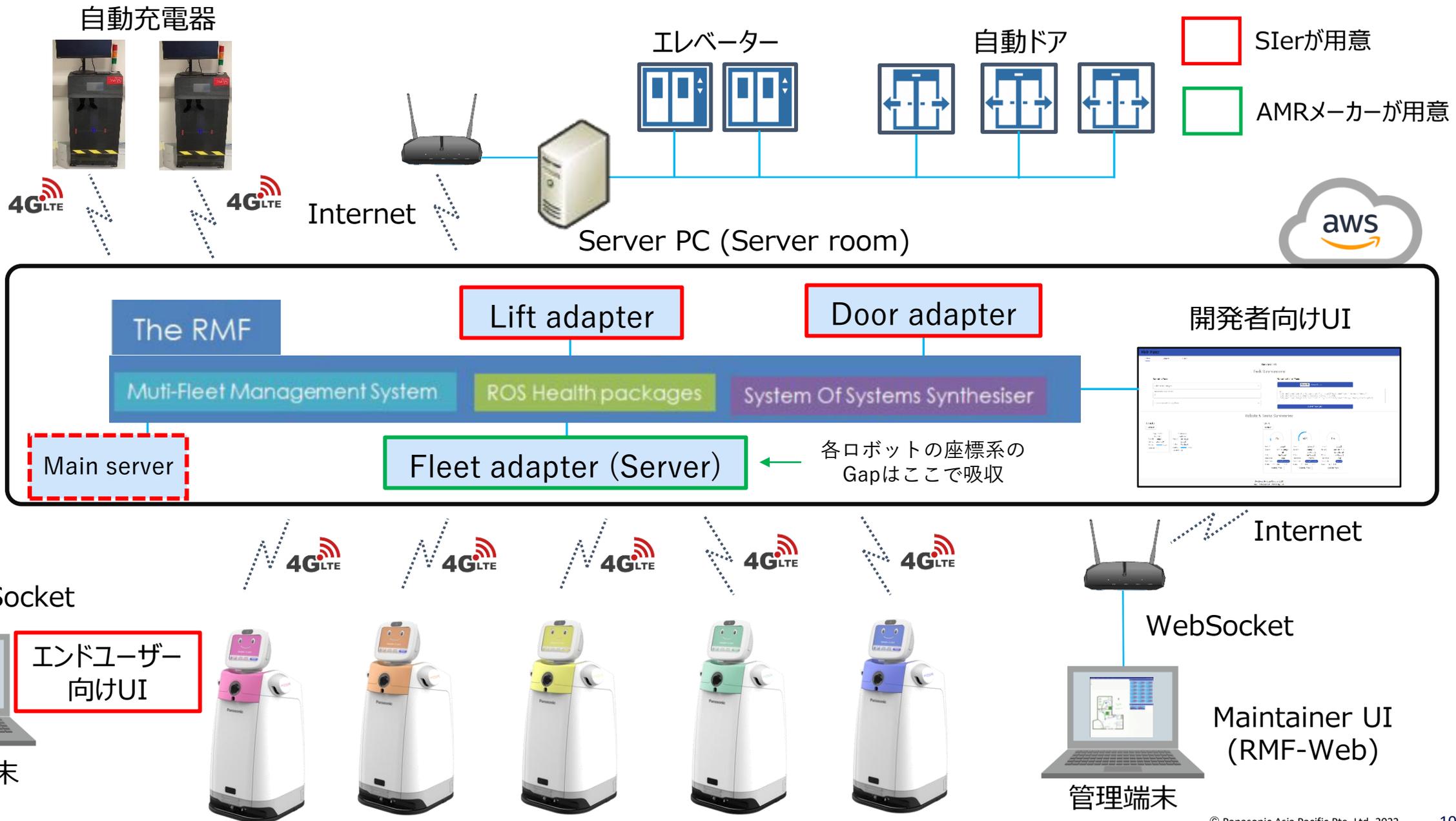
この2台のどちらか一方といったリクエストは不可能

# 外部指令による全ロボットの退避位置への移動

火災報知器・コードブルー発令時にロボットを退避位置に退避させることが可能  
(注意) 2台のロボットが同じ位置に退避する場合はスタックする事に留意



# RMFを用いたAMRの制御システム構成 (例)



# 開発者向けUI

## 開発者向けUIはRMFにタスクを送る機能がメイン とてもエンドユーザー向けではない...

### RMF Panel

Office Airport Clinic

Time is: 0:1:37

#### Task Submissions

##### Submit a Task

Select a request type

Set start time (mins from now)

0

Choose an evaluator (optional)

##### Submit a List of Tasks

Choose file office\_tasks.json

```
eg. [
  {"task_type": "Loop", "start_time": 0, "description": {"num_loops": 5, "start_name": "coe", "finish_name": "lounge"}},
  {"task_type": "Delivery", "start_time": 0, "description": {"option": "coke"}},
  {"task_type": "Loop", "start_time": 0, "description": {"num_loops": 5, "start_name": "cubicle_2", "finish_name": "supplies"}}
]
```

SUBMIT TASK LIST

#### Robots & Tasks Summaries

##### Robots

REFRESH

<p>tinyRobot1</p> <p>tinyRobot</p> <p>Task ID Loop0</p> <p>Status Moving-2</p> <p>Battery  99.88%</p> <p>Location L1</p>	<p>tinyRobot2</p> <p>tinyRobot</p> <p>Task ID Delivery1</p> <p>Status Moving-2</p> <p>Battery  99.89%</p> <p>Location L1</p>
--	--

##### Tasks

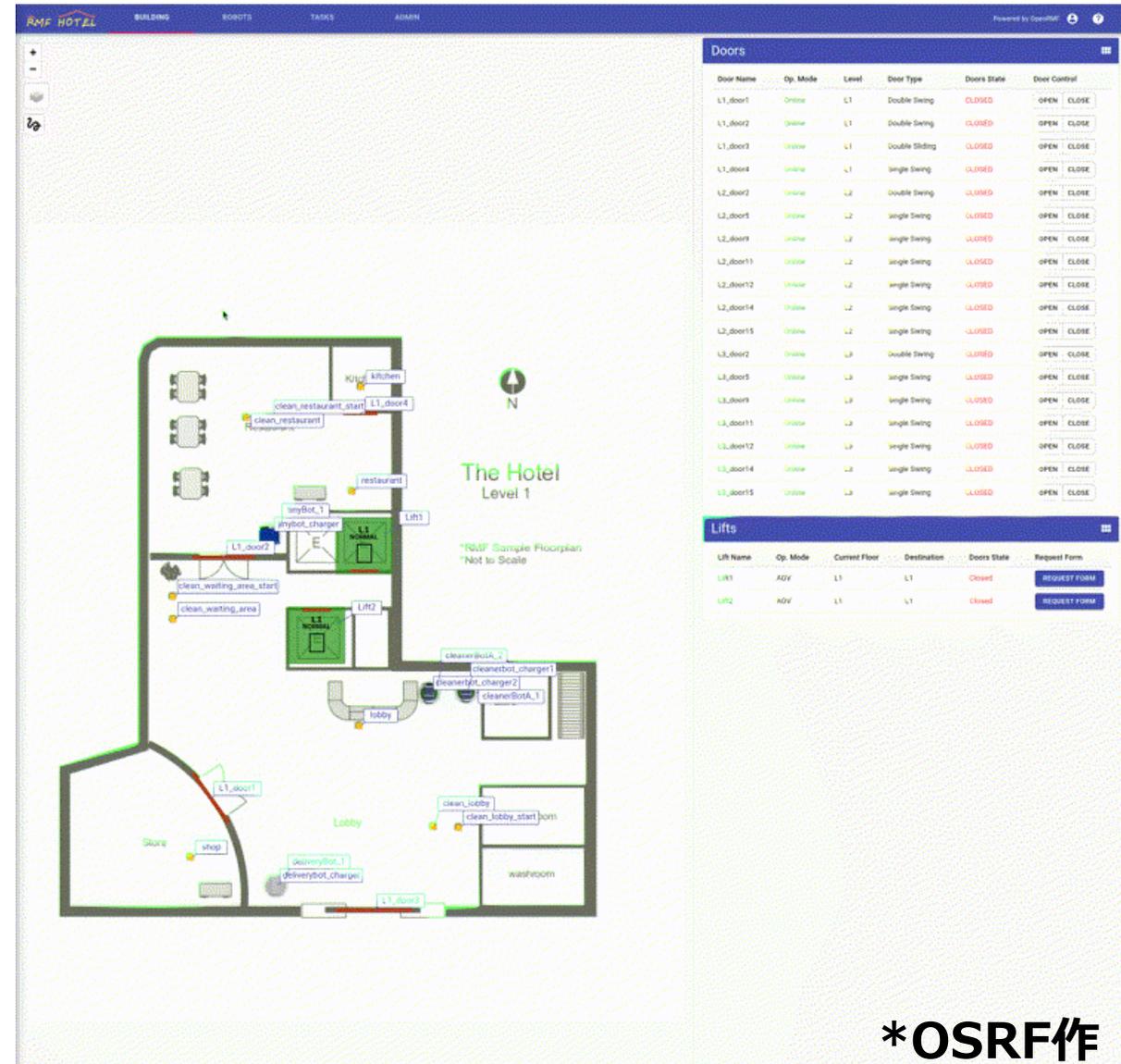
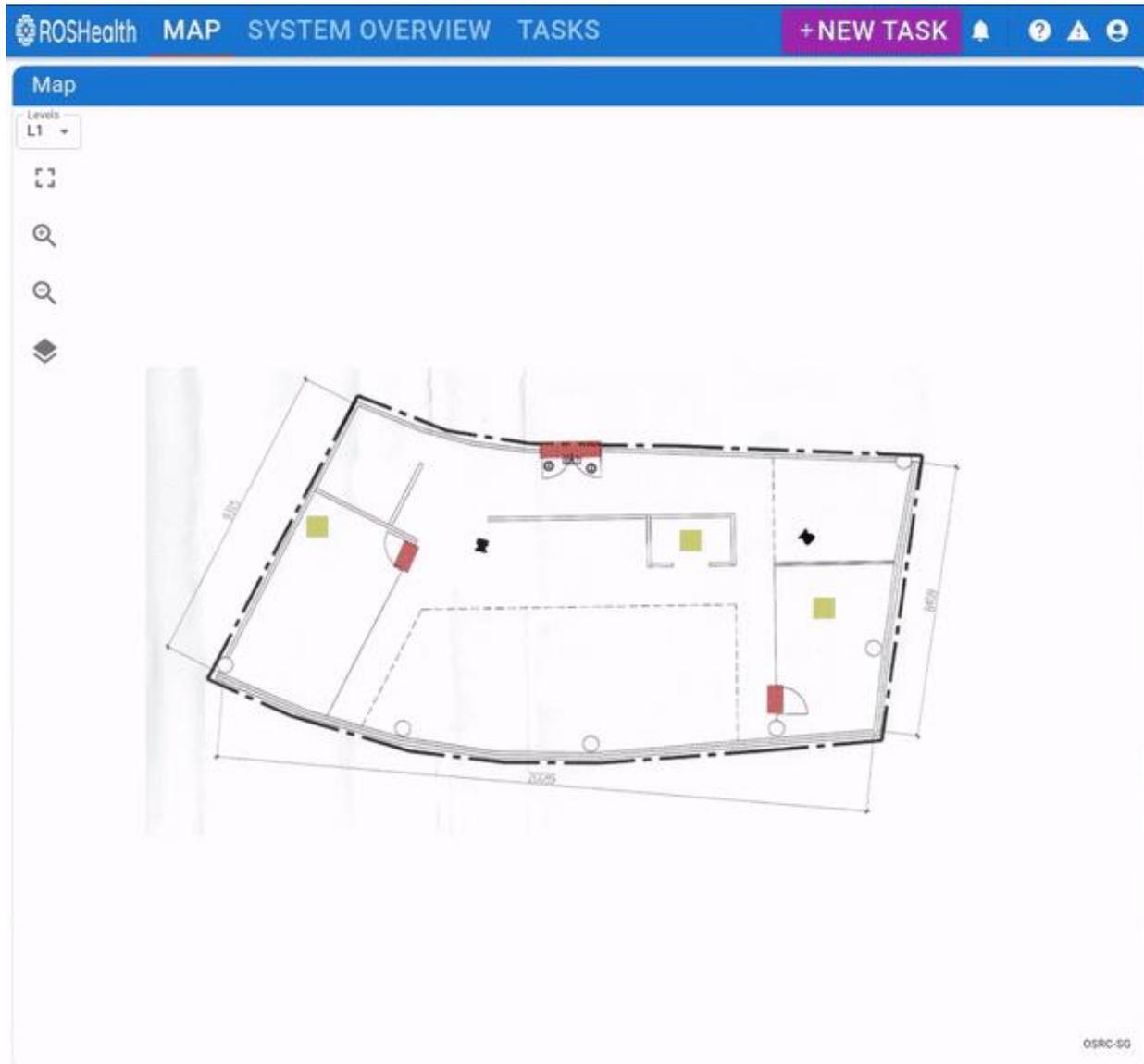
REFRESH

<p>7%</p> <p>Task ID Loop0</p> <p>Details coe --&gt; lounge x5</p> <p>Robot tinyRobot1</p> <p>Task Type Loop</p> <p>Task State <b>Active/Executing</b></p> <p>Start: 55 End: 585</p> <p>CANCEL TASK</p>	<p>54%</p> <p>Task ID Delivery1</p> <p>Details pantry --&gt; hardware_2</p> <p>Robot tinyRobot2</p> <p>Task Type Delivery</p> <p>Task State <b>Active/Executing</b></p> <p>Start: 58 End: 130</p> <p>CANCEL TASK</p>	<p>0%</p> <p>Task ID Loop2</p> <p>Details cubicle_2 --&gt; supplies x5</p> <p>Robot tinyRobot2</p> <p>Task Type Loop</p> <p>Task State <b>Queued</b></p> <p>Start: 692 End: 0</p> <p>CANCEL TASK</p>
---	--	--

# RMFが用意しているエンドユーザー向けUI (RMF-Web)

## テンプレート

## カスタマイズ例\*



\*OSRF作

# 制作しなければならないUIの例

お客様のロボットの運用に応じてエンドUIはデザインが様々な  
殆どの案件でSIerがスクラッチで開発する必要がある考える  
一部の機能はRMF-Webを参考にすることも可能

The screenshot shows the HOSPI web interface. At the top, there are tabs for 'Fetch Setting', 'Schedule Setting', and 'FMS Information'. Below is a 'Status' table with columns for Robot, Name, Status, Battery, Connection, Door, Reservation, Task ID, Cart, Task, and Error. Two rows are visible: one for 'hospi signage' (STATUS\_ACTION...) and one for 'hospi' (STATUS\_IDLE). Below the table is a 'Call Robot' form with dropdowns for Robot (hospi), Place (piimo\_1\_dock), and Task (delivery), and a 'Call' button. A red box highlights the 'Call Robot' form and the 'Robot' column of the status table, with the text 'ロボットのステータス' (Robot Status) next to it. A red box highlights the 'Call' button with the text '呼出機能' (Call Function). A red box highlights the 'Robot' dropdown with the text 'スマホ/タブレット切替' (Smartphone/Tablet Switch). An arrow points from the 'Call' button to a smartphone mockup showing the same interface on a mobile device.

Robot	Name	Status	Battery	Connection	Door	Reservation	Task ID	Cart	Task	Error
hospi signage	hospi signage	STATUS_ACTION...	76%	↑	🔒	reserved	4		call	XXXX
hospi	hospi	STATUS_IDLE	70%	↑	🔒	free	362		call	XXXX

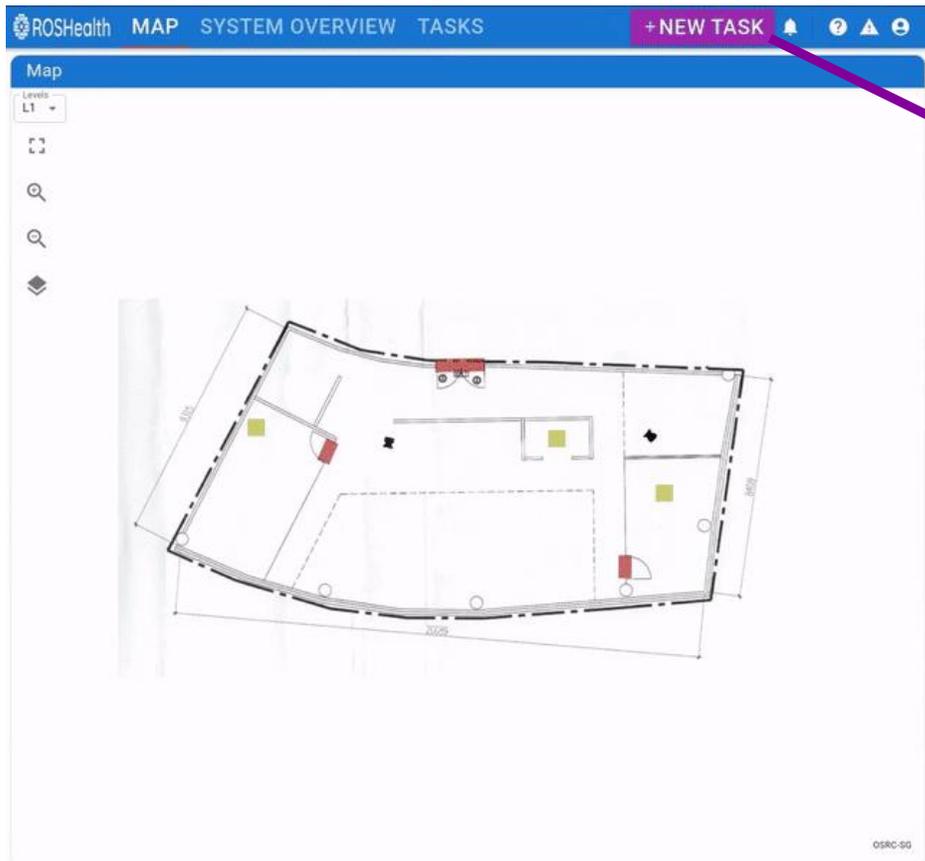
The screenshot shows the HOSPI web interface with a 3D map of the robot's environment. The map is labeled 'L1' and shows various rooms, desks, and chairs. A robot is visible on the map, and there are several task markers. The interface includes tabs for 'MAP', 'INFRASTRUCTURE', and 'TASKS'. A 'Robot Selection' dropdown is set to 'hospi', and there are buttons for 'CALL ROBOT' and '+ NEW TASK'. The map shows a layout with desks, chairs, and a robot moving through the space.

ロボットの地図上での表示機能は  
RMF-Webの機能を使えば簡単に実現可能

ロボットの呼出機能などはお客様に  
使いやすい用にスクラッチで開発

# (参考) RMF-WebのCreate Task

RMF-web自身にはタスク入力画面もある  
(スケジュール設定により自動割り当ても存在)



### Create Task

**Favorite tasks**

Task Category

Start Time

Priority

Place Name: A007

# RMFが用意している運用ログ

**HOSPI**
MAP    INFRASTRUCTURE    **TASKS**
Robot Selection < --Select Robot-- >
CALL ROBOT    + NEW TASK    64

### Tasks

QUEUE
SCHEDULE

Date	Requester	ID	Category	Assignee	Start Time	End Time	State
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 00:00:00	2024/9/20 00:00:00	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 18:31:37	2024/9/20 18:32:37	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:53:37	2024/9/20 16:54:36	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:51:30	2024/9/20 16:52:41	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:49:01	2024/9/20 16:51:30	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:48:36	2024/9/20 16:48:36	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:47:36	2024/9/20 16:48:36	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:47:36	2024/9/20 16:47:36	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:46:00	2024/9/20 16:47:36	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:44:33	2024/9/20 16:46:00	completed
2024/9/20	panauser	patrol.dis...	Patrol	hospi	2024/9/20 16:43:50	2024/9/20 16:44:33	completed

1-10 of 11    <    >

### Map

# RMFがUI用に用意しているAPI

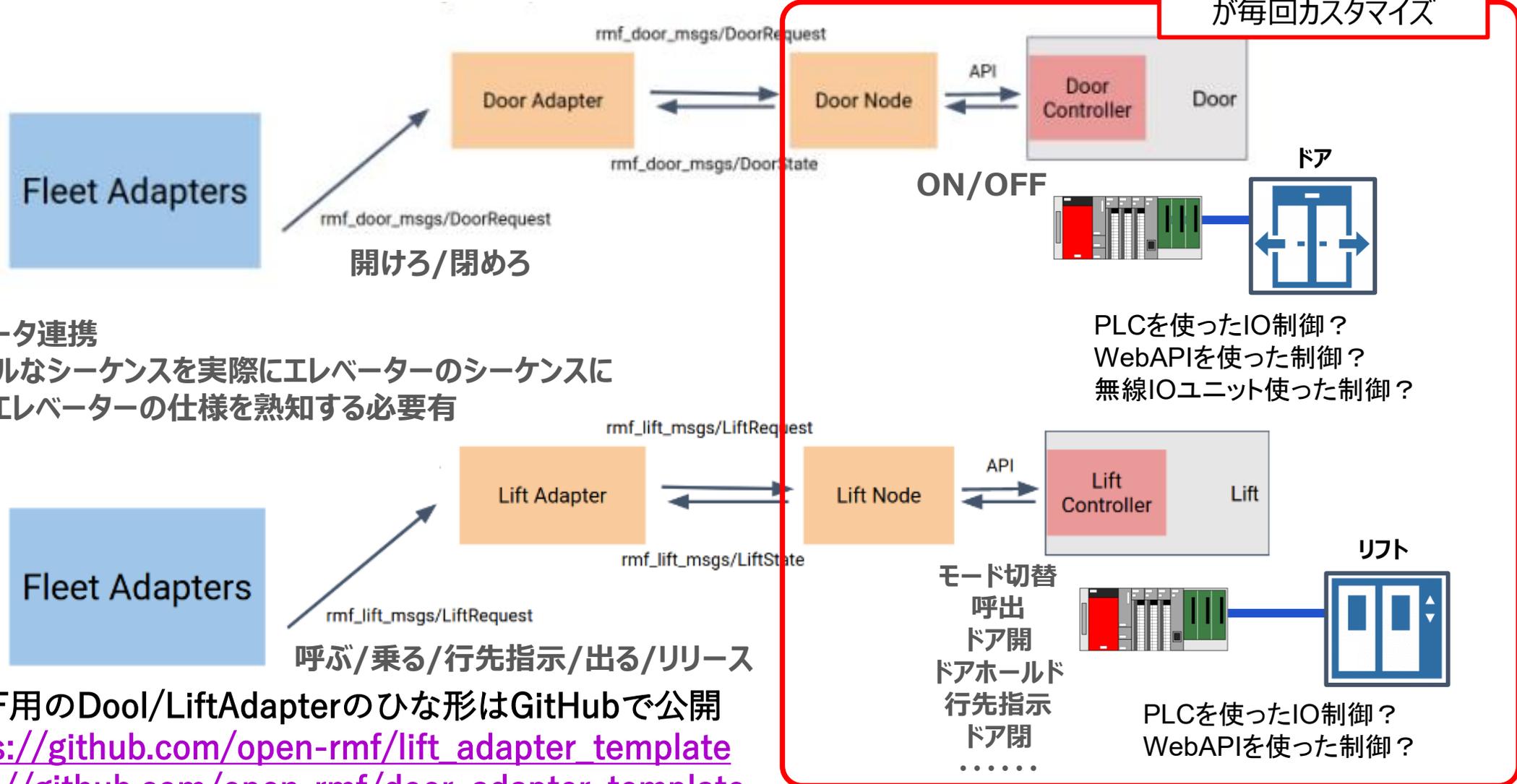
RMF-webは下記のAPIを使用している  
つまり、このAPIを使って同じような機能のAIを自分専用の開発言語で作成する事も可能  
自分達のシステムと連携させることも容易

default	Tasks	Dispensers
<b>GET</b> /socket.io Socket.io endpoint	<b>GET</b> /tasks/{task id}/request Get Task Request	<b>GET</b> /dispensers Get Dispensers
<b>GET</b> /user Get User	<b>GET</b> /tasks Query Task States	<b>GET</b> /dispensers/{guid}/state Get Dispenser State
<b>GET</b> /permissions Get Effective Permissions	<b>GET</b> /tasks/{task id}/state Get Task State	<b>GET</b> /dispensers/{guid}/health Get Dispenser Health
<b>GET</b> /time Get Time	<b>GET</b> /tasks/{task id}/log Get Task Log	
<b>Alerts</b>	<b>POST</b> /tasks/activity discovery Post Activity Discovery	<b>Ingestors</b>
<b>GET</b> /alerts Get Alerts	<b>POST</b> /tasks/cancel task Post Cancel Task	<b>GET</b> /ingestors Get Ingestors
<b>POST</b> /alerts Create Alert	<b>POST</b> /tasks/dispatch task Post Dispatch Task	<b>GET</b> /ingestors/{guid}/state Get Ingestor State
<b>GET</b> /alerts/{alert id} Get Alert	<b>POST</b> /tasks/robot task Post Robot Task	<b>GET</b> /ingestors/{guid}/health Get Ingestor Health
<b>POST</b> /alerts/{alert id} Acknowledge Alert	<b>POST</b> /tasks/interrupt task Post Interrupt Task	
<b>Beacons</b>	<b>POST</b> /tasks/kill task Post Kill Task	<b>Fleets</b>
<b>GET</b> /beacons Get Beacons	<b>POST</b> /tasks/resume task Post Resume Task	<b>GET</b> /fleets Get Fleets
<b>POST</b> /beacons Save Beacon State	<b>POST</b> /tasks/rewind task Post Rewind Task	<b>GET</b> /fleets/{name}/state Get Fleet State
<b>GET</b> /beacons/{beacon id} Get Beacon	<b>POST</b> /tasks/skip phase Post Skip Phase	<b>GET</b> /fleets/{name}/log Get Fleet Log
<b>Building</b>	<b>POST</b> /tasks/task discovery Post Task Discovery	
<b>GET</b> /building map Get Building Map	<b>POST</b> /tasks/undo skip phase Post Undo Skip Phase	
<b>Doors</b>	<b>POST</b> /scheduled tasks Post Scheduled Task	<b>Admin</b>
<b>GET</b> /doors Get Doors	<b>GET</b> /scheduled tasks Get Scheduled Tasks	<b>GET</b> /admin/users Get Users
<b>GET</b> /doors/{door name}/state Get Door State	<b>GET</b> /scheduled tasks/{task id} Get Scheduled Task	<b>POST</b> /admin/users Create User
<b>GET</b> /doors/{door name}/health Get Door Health	<b>DELETE</b> /scheduled tasks/{task id} Del Scheduled Tasks	<b>GET</b> /admin/users/{username} Get User
<b>POST</b> /doors/{door name}/request Post Door Request	<b>PUT</b> /scheduled tasks/{task id}/clear Del Scheduled Tasks Event	<b>DELETE</b> /admin/users/{username} Delete User
<b>Lifts</b>	<b>POST</b> /scheduled tasks/{task id}/update Update Schedule Task	<b>POST</b> /admin/users/{username}/make admin Make Admin
<b>GET</b> /lifts Get Lifts	<b>POST</b> /favorite tasks Post Favorite Task	<b>POST</b> /admin/users/{username}/roles Add User Role
<b>GET</b> /lifts/{lift name}/state Get Lift State	<b>GET</b> /favorite tasks Get Favorites Tasks	<b>PUT</b> /admin/users/{username}/roles Set User Roles
<b>GET</b> /lifts/{lift name}/health Get Lift Health	<b>DELETE</b> /favorite tasks/{favorite task id} Delete Favorite Task	<b>DELETE</b> /admin/users/{username}/roles/{role} Delete User Role
<b>POST</b> /lifts/{lift name}/request Post Lift Request		<b>GET</b> /admin/roles Get Roles
		<b>POST</b> /admin/roles Create Role
		<b>DELETE</b> /admin/roles/{role} Delete Role
		<b>GET</b> /admin/roles/{role}/permissions Get Role Permissions
		<b>POST</b> /admin/roles/{role}/permissions Add Role Permission
		<b>POST</b> /admin/roles/{role}/permissions/remove Remove Role Permission

# RMFを使ったインフラ連携

各インフラに合わせてSIerによってノードのカスタマイズが必要  
インフラ工事の経験が無い企業には敷居が高い

ここでだけは現場でSier  
が毎回カスタマイズ

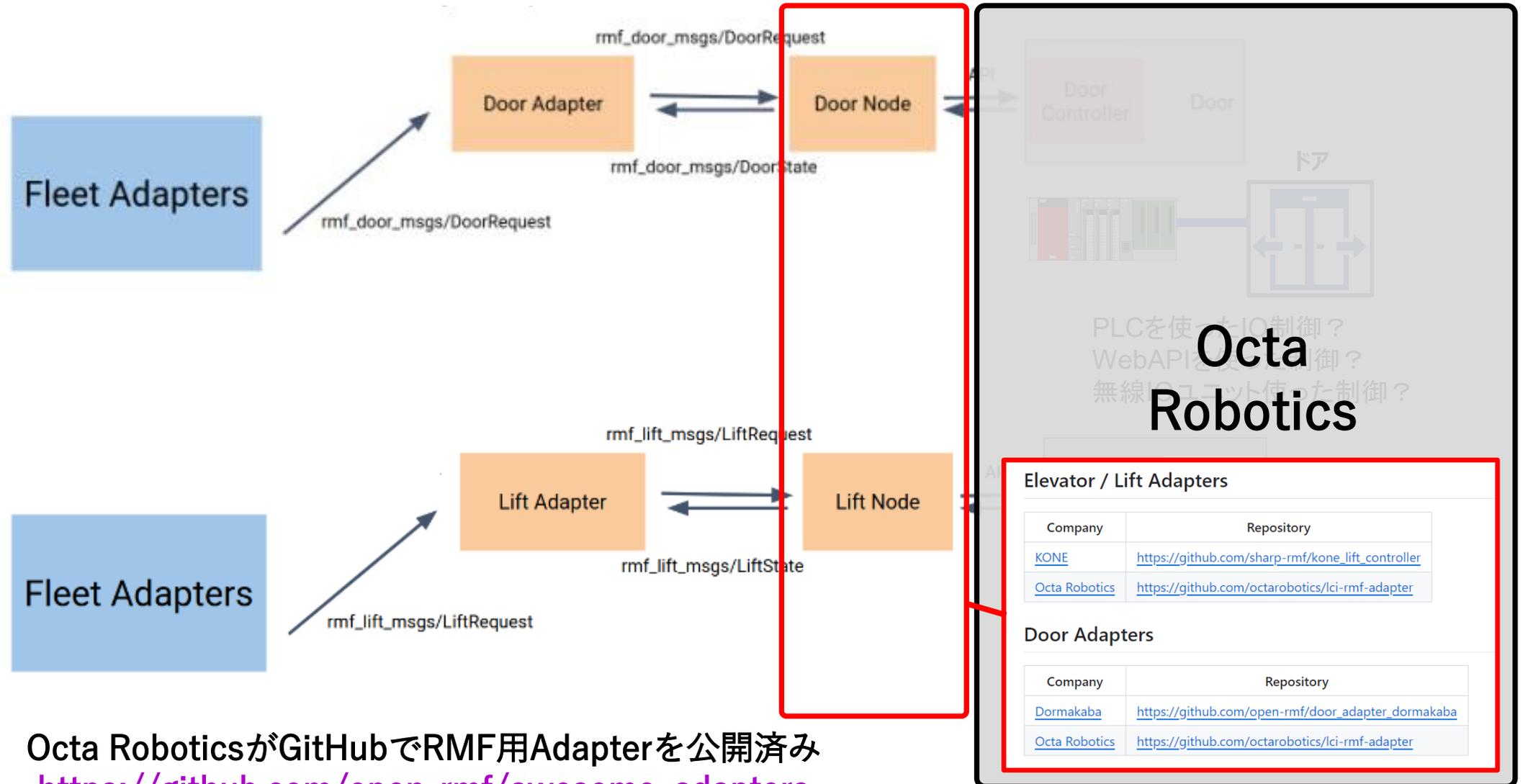


注意：エレベータ連携  
RMFのシンプルなシーケンスを実際にエレベーターのシーケンスに  
適合するのはエレベーターの仕様を熟知する必要有

RMF用のDoor/LiftAdapterのひな形はGitHubで公開  
[https://github.com/open-rmf/lift\\_adapter\\_template](https://github.com/open-rmf/lift_adapter_template)  
[https://github.com/open-rmf/door\\_adapter\\_template](https://github.com/open-rmf/door_adapter_template)

# RMFを使ったインフラ連携

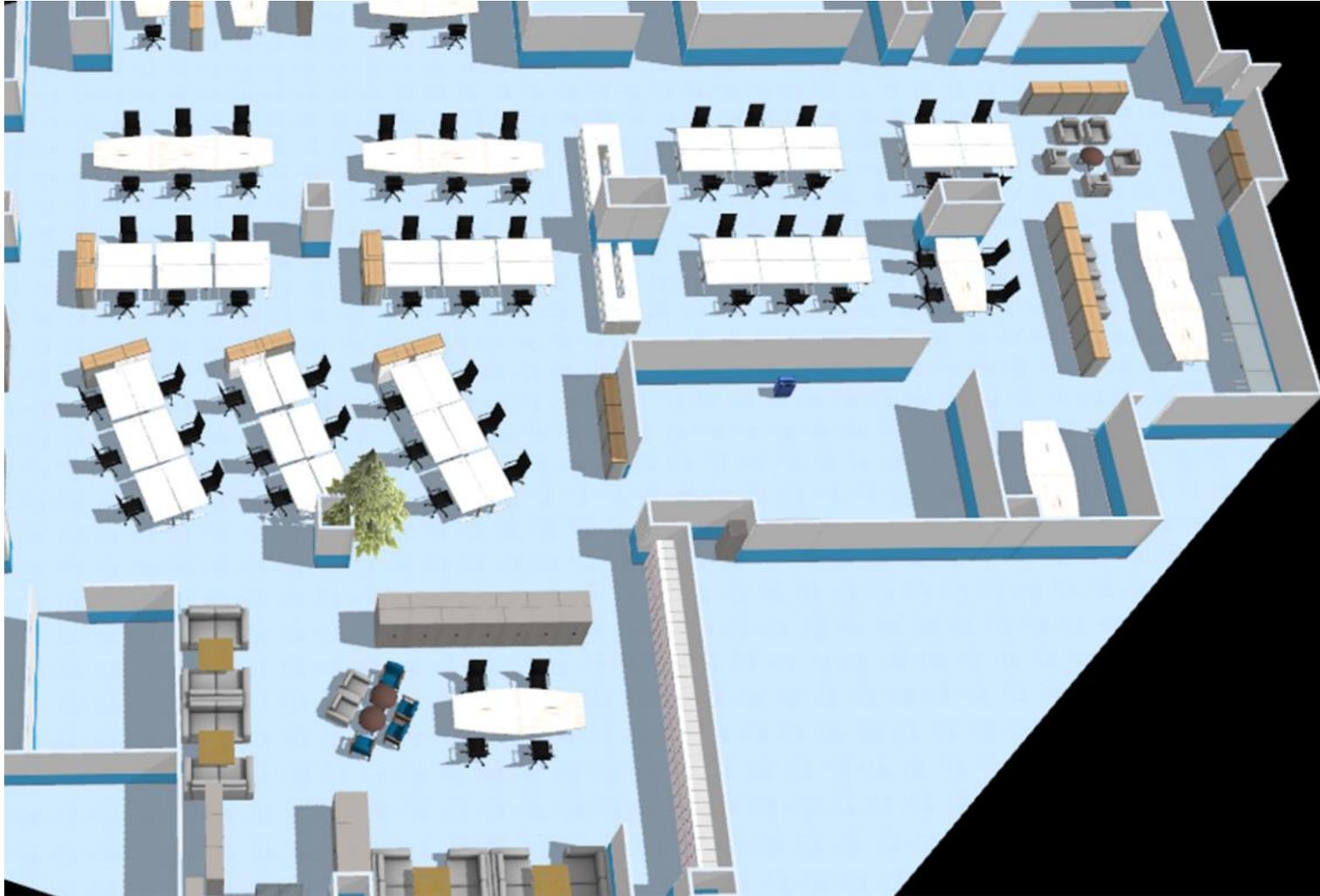
Octa Roboticsに依頼すれば誰でもRMFでロボットとインフラ連携が簡単に実現可能  
RFA (Robot Friendly Asset Promotion Association) 規格にも対応



Octa RoboticsがGitHubでRMF用Adapterを公開済み  
[https://github.com/open-rmf/awesome\\_adapters](https://github.com/open-rmf/awesome_adapters)

# (参考) RMFを使った3次元シミュレーション環境の作成

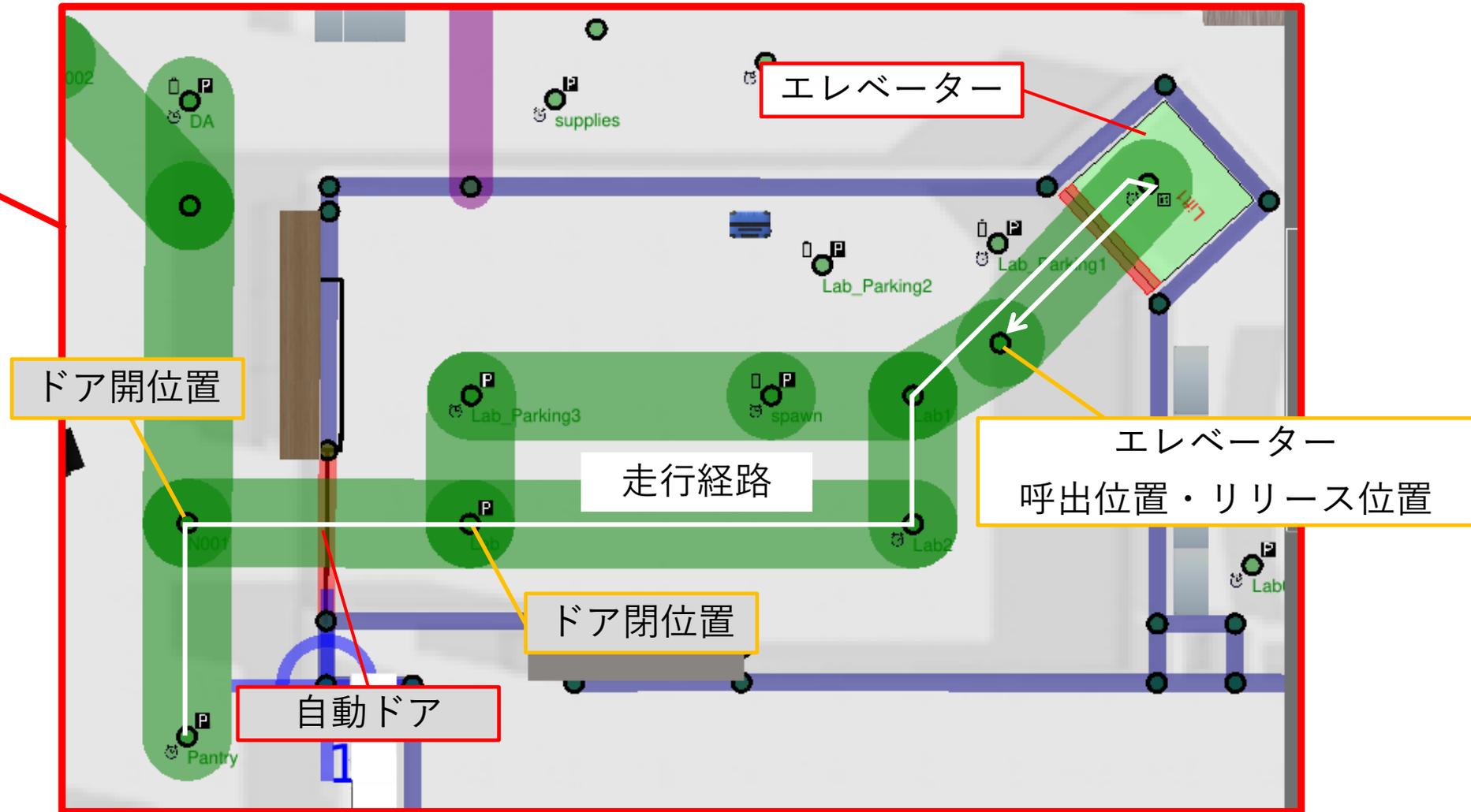
(例) シンガポールの自社オフィスを模擬したシミュレーション環境  
我々のオフィスでSLAMで作成した地図をベースに評価環境を作成



3時間程度で作成が可能です。もちろん、複数階の建物 エレベーター 自動ドア も設置可能です。

# RMF制御時の自動ドア、エレベーター連携

エレベーターのシミュレーションにOcta Roboticsのシミュレーターを使用  
RMF上に該当場所にエレベーターを設置し、Octa RoboticsのLift Adapterを起動するだけ





The image features the Panasonic logo in white, bold, sans-serif font, centered on a dark blue background. The background is decorated with various geometric elements: a large, light blue semi-circle on the left side; a grid of thin white lines; a bright cyan square in the top right corner; and a medium blue square in the middle right area. The overall aesthetic is modern and corporate.

**Panasonic**