<오동동>
<Mobile Robot Simulator – ADD_ON>
설계 산출물

[요 약]

Mobile Robot Controller 의 ADD_ON 시스템의 설계 산출물을 기술한다.
주요 산출물로

1. Package Diagram

2. Class Diagram

3. Interaction Diagram

4. State Machine Diagram

을 작성하였다.

1 개 요

1.1 목 적

본 문서에서는 Mobile Robot Controller 시스템에 대한 설계 산출물을 기술한다.

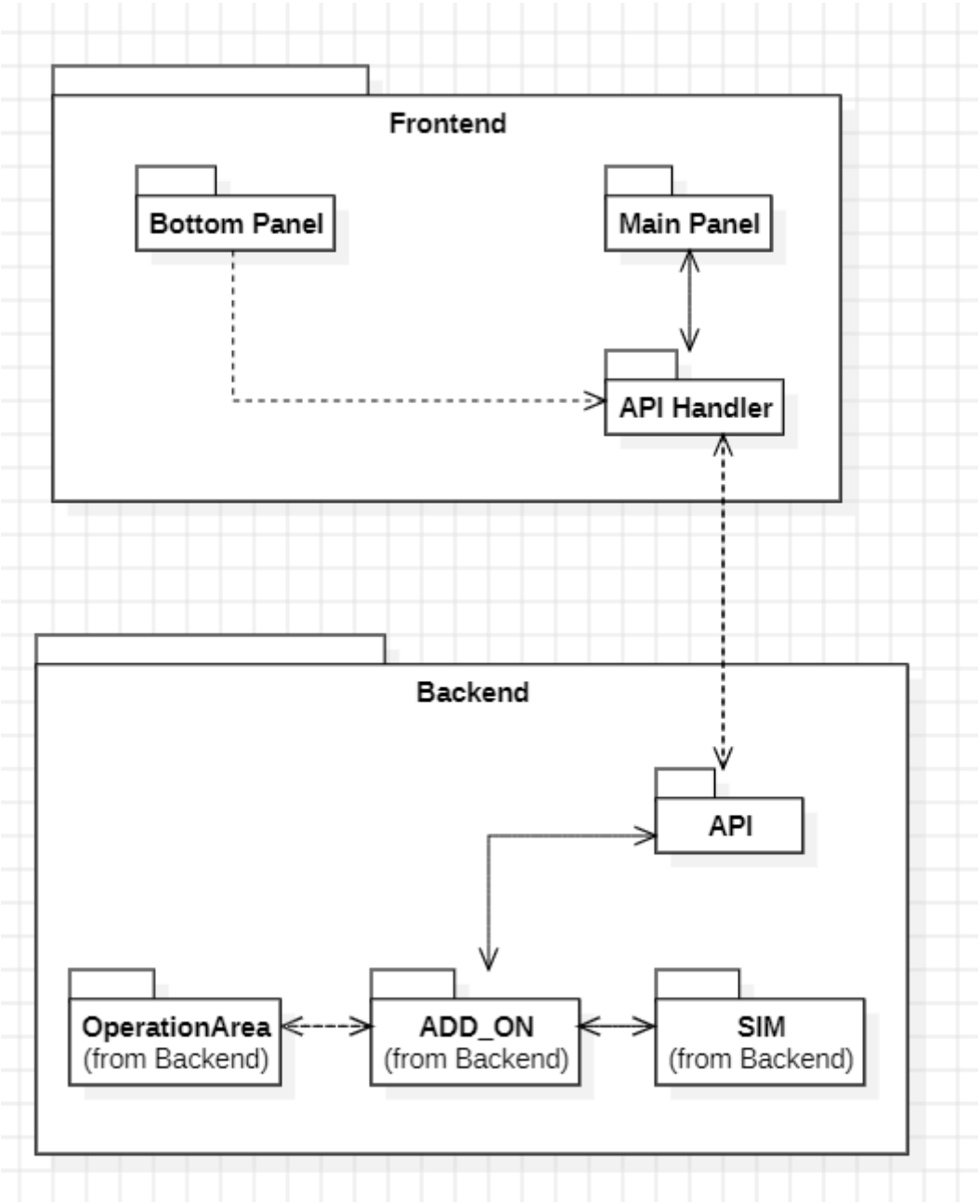
1. **Package Diagram** 을 활용하여 시스템의 구조를 가시적으로 표현한다.

2. **Class Diagram** 을 작성하여 시스템이 가지는 클래스와, 그들의 관계를 표현한다.

3. Object 간에 발생하는 동적인 교류들을 **Interaction Diagram** 을 통해 표현한다.

4. 특정 Object 의 상태 변화를 **State Chart Diagram** 으로 표현한다.

2. Package Diagram



전체 시스템은 Frontend 와 Backend 로 나누어지며, 로봇의 동작과 관련되는 시스템은 Backend 의 4 가지 서브 시스템으로 구성된다.

1. API 시스템

완성된 산출물은 웹 페이지로 구현된다. 따라서, 사용자(Operator)의 입력(OperationArea 초기화 및 재난 지형 업데이트)를 api 를 통하여 전달

받는다.
전달 받은 api 의 JSON 파일을 읽어 ADD_ON 시스템이 사용하는 언어로 해석하여 전달한다. python 의 웹프레임워크 Flask 를 사용한다.

2. ADD_ON 시스템

API 시스템과 상호작용 하여 사용자의 입력을 받고, 경로 계산등의 연산을 다시 사용자에게 전달한다.

또한, 경로 계산을 위해 OperationArea 시스템과 SIM 시스템과 통신한다.

로봇의 동작을 지시하기 위해 SIM 시스템과 상호작용을 하며, 이러한 과정에서 업데이트 된 정보를 갱신하기 위해 OperationArea 시스템과 통신한다.
Python3 언어로 작성된다.

3. SIM 시스템

ADD_ON 시스템과 상호작용한다.

실제 재난 지역을 탐사할 로봇을 표현하는 시스템이다. ADD_ON 시스템과의 상호작용 하여 로봇을 움직인다.

Python3 언어로 작성된다.

4. OperationArea 시스템

재난 지역 모델을 표현하는 시스템이다.

ADD_ON 시스템으로부터 재난 지역 모델의 초기화, 변경을 지시받는다. 웹페이지에 그릴 재난 지역 모델의 데이터를 전달하기도 한다.

Python3 언어로 작성된다.

3 “ADD_ON” 서브시스템 세부 설계

ADD_ON 시스템은 OperationArea 시스템에 접근해서 재난 지역 모델의 데이터를 받아올 수 있고, SIM 시스템의 로봇 동작을 지시할 수 있다.

ADD_ON 시스템은 여러 기능들로 구성되며, 각 기능들은 저마다 외부 서브시스템들과 상호작용한다.

[UpdateOperationArea]

재난 지역 모델의 데이터를 변경한다.

[GeneratePath]

로봇이 탐사할 경로를 결정한다.

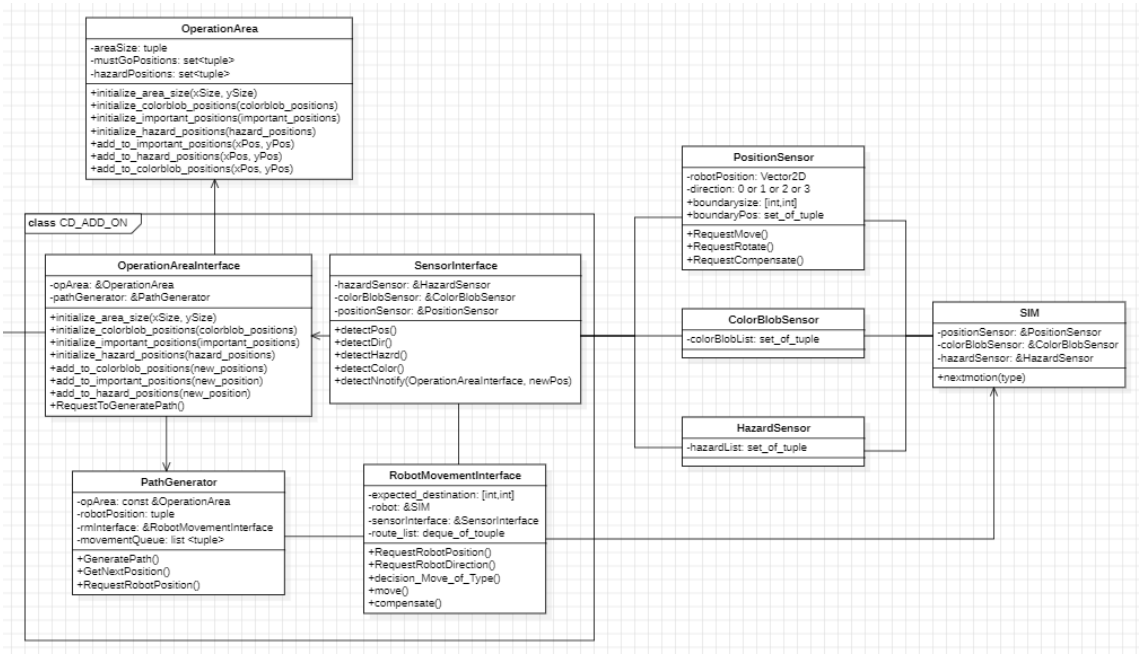
[Move]

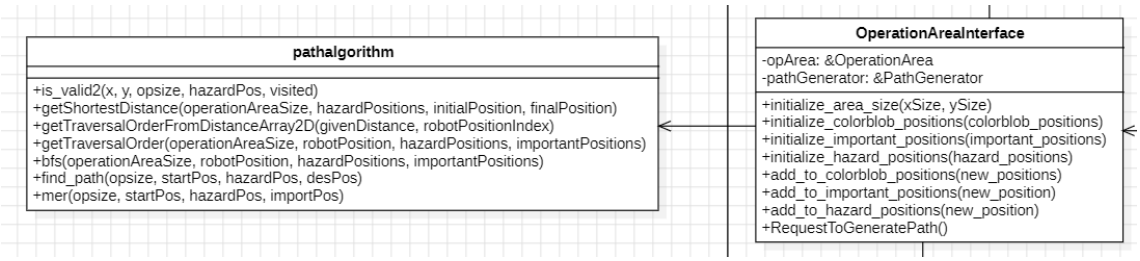
목적지를 향해 RobotMovementInterface 가 로봇을 이동시킨다.

[Compensate]

로봇이 오작동 할 때 RobotMovementInterface 가 목적지로 로봇을 이동시킨다.

3.1 Class Diagram





[SIM]

재난 지역을 탐사할 로봇이다.

세 센서 PositionSensor, ColorBlobSensor, HazardSensor 를 가지며 센서들은 SensorInterface 를 통해 ADD_ON 시스템과 통신한다.

[PositionSensor]

로봇의 현재 위치를 감지하는 센서이다.

[ColorBlobSensor]

로봇의 상하좌우 1 칸이 Color Blob 인지 탐지하는 센서이다.
실제 구현은, 지도의 불완전한 정보를 사용자의 음성인식으로 입력받아, 이러한 정보들을 tuple 들의 set 으로 저장해 두었다가, 현재 위치와 비교를 하여 탐지한다.

[HazardSensor]

로봇의 전방 1 칸이 Hazard 인지 탐지하는 센서이다.
실제 구현은, 지도의 불완전한 정보를 사용자의 음성인식으로 입력받아, 이러한 정보들을 tuple 들의 집합으로 저장해두었다가, 현재 위치와 비교해 탐지한다.

[SensorInterface]

ADD_ON 시스템에서 로봇의 센서들과 소통하는 Interface 이다.
로봇의 세 센서들과의 상호작용은 다음과 같다:

1. HazardSensor 로부터 위험 지점을 탐지한 경우, OperationAreaInterface 에 재난 지역 모델 변경을 지시한다.

2. ColorBlobSensor 로부터 ColorBlob 지역을 탐지한 경우, OperationAreaInterface 에 재난 지역 모델 변경을 지시한다.

3. 로봇의 현재 위치가 필요한 경우, PositionSensor 에 질의하여 로봇의 현재 위치를 알아낸다.

[RobotMovementInterface]

로봇의 이동을 전담하는 Interface 이다. 이를 통하여 ADD_ON 은 SIM 에

동작을 지시함으로써 통신한다.
생성된 route_list 로 다음 동작을 결정하고 이를 SIM 에게 지시한다.
로봇의 오동작을 보정하는 작업을 수행한다.

[PathGenerator]

로봇이 움직일 경로를 계산, 결정하는 객체로 시스템에 오로지 하나 존재한다.
재난 지역 모델이 변경될 때 마다 호출되며 경로를 재계산한다.

[OperationArea]

재난 지역 모델을 나타내는 객체로 시스템에 오로지 하나 존재한다.
ADD_ON 시스템과는 오로지 OperationAreaInterface 로만 소통한다.

[OperationAreaInterface]

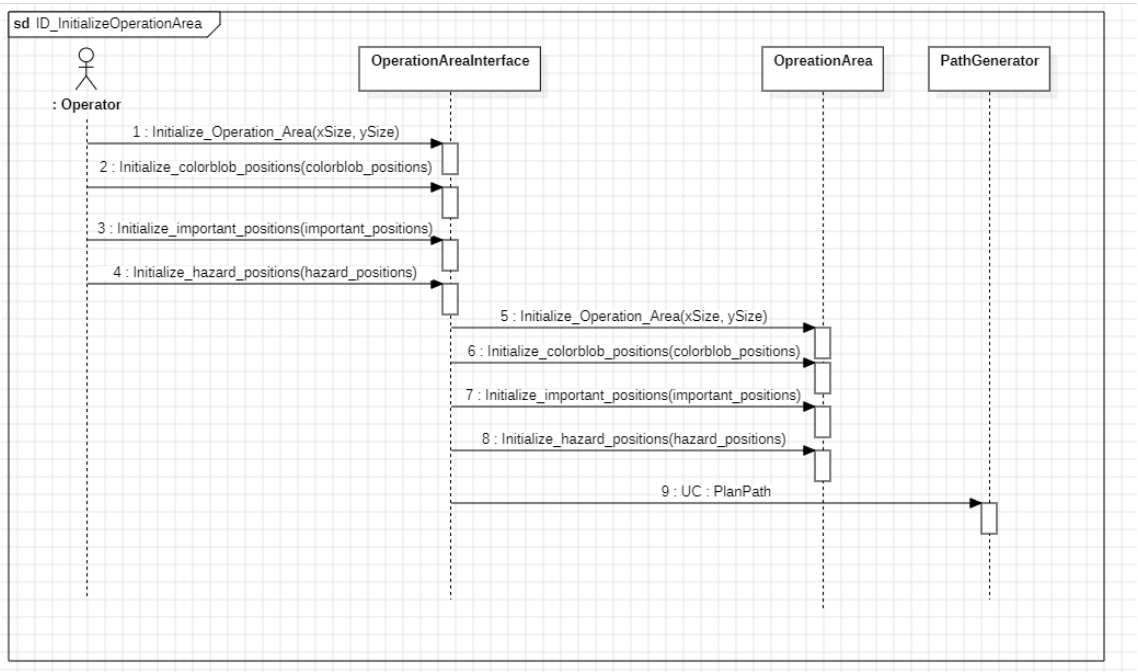
ADD_ON 에서 재난 지역 모델과 소통하는 Interface 이다.
재난 지역 모델과는

1. 초기화를 지시한다.

2. 재난 지역 모델의 데이터를 변경한다.

의 상호작용을 한다.

4.1 Interaction Diagram (InitializeOperationArea)

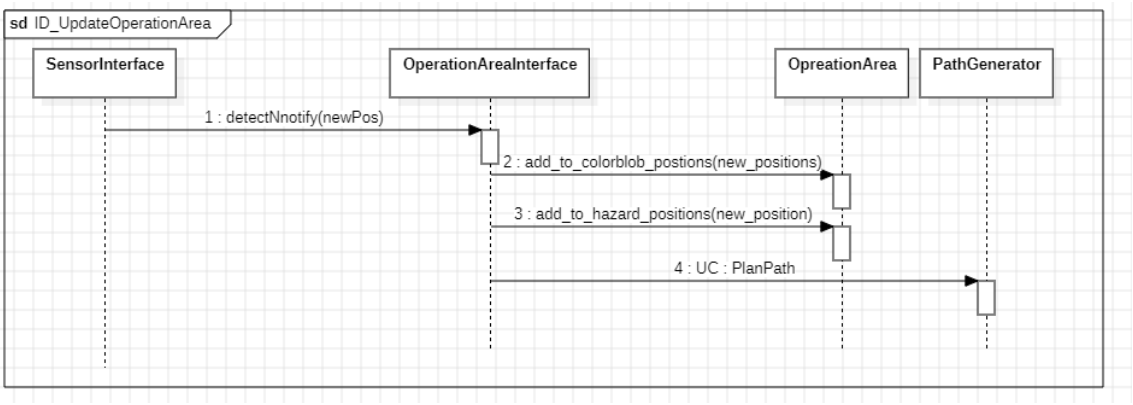


OperationAreaInterface 는 초기화 단계에서 사용자가 입력한 재난 지역 모델의 정보를 전달한다. 전체 시스템에서 OperationArea 는 오로지 하나만 존재한다.

Initialize 를 통해 재난 지역 모델의 크기와 predefined 된 colorblob, important, hazard positions 들을 초기화하여 OperationArea 정보를 저장한다.
이때 매개변수로 전달되는 positions 들의 자료형은 (int, int)로 된 touple 들의 set 이다.

초기화 작업이 마친 이후, 재난 지역 모델의 데이터를 PathGenerator 에게 전달하며 경로 계산을 지시한다.

4.2 Interaction Diagram (UpdateOperationArea)



해당 교류도가 표현하는 상황은 다양한 요소들에 의해 발생할 수 있다.

1. HazardSensor 가 전방에 기록되지 않은 위험 지역을 탐지하는 경우

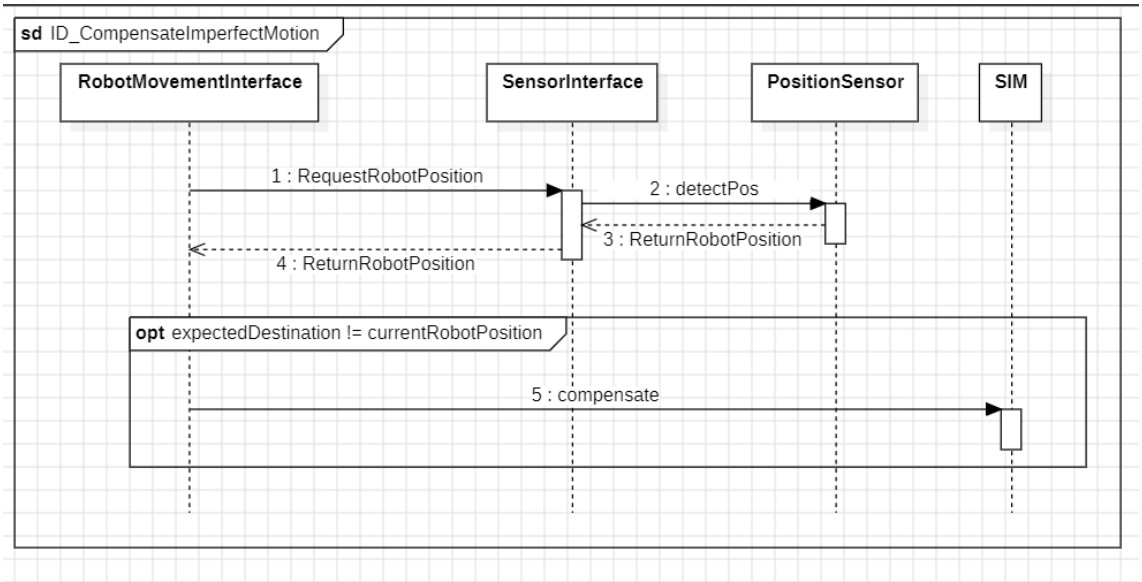
2. ColorBlobSensor 가 상하좌우 1 칸에 기록되지 않은 중요 지점을 탐지하는 경우

위 이벤트들이 발생할 때, OperationAreaInterface 에 정보가 전달되며 이를 토대로 재난 지역 모델 변경 및 경로 계산 작업이 진행된다.

이때, SensorInterface 는 OpreationAreaInterface 에게 detectNnotify 로 발견된 정보들을 newPos 이름의 list 로 전달된다. 이 리스트는 tuple 들로 구성되어 있으며, 튜플 안에는 detect 된 spot 의 타입('C' or 'H')와 그 위치들로 구성된다.

이 정보를 해석하여, OpreationArea 를 업데이트를 하며, PathGenerator 에게 경로를 생성을 요청하는 plan_path 상황(interaction diagram 4.4)이 실행된다.

4.3 Interaction Diagram (CompensateImperfectMotion)

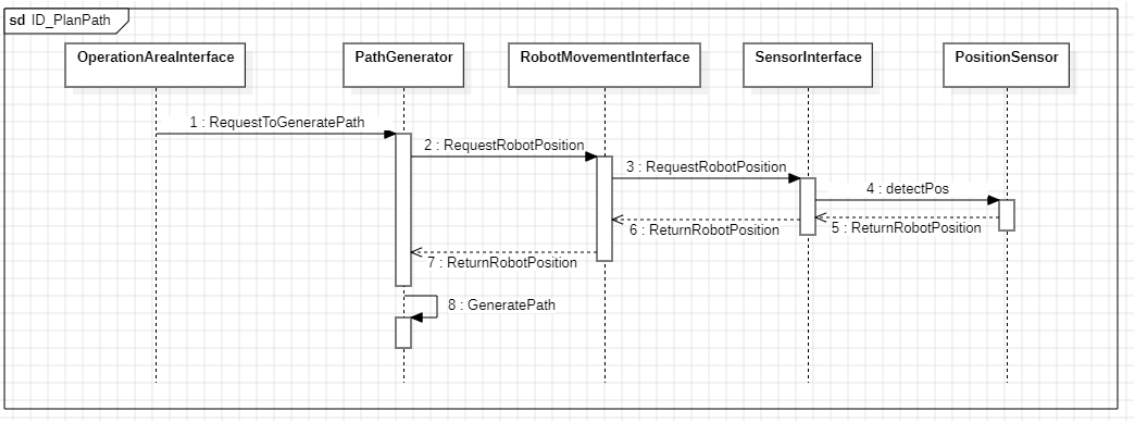


RobotMovementInterface 는 로봇이 움직이기 전, 위치를 검사하며 보정이 필요할 경우 보정 작업을 수행한다. 위 교류도는 위치 보정 작업을 나타내는 다이어그램이다.

RequestRobotPosition() 은 현재 로봇의 위치를 질의하는 메소드이다. 이 질의를 통해, SensorInterface 는 PositionSensor 에게 detectPos 를 통해 실제 위치를 반환받아 Pathgenerator 에게 값을 전달한다.

Compensate() 은 오작동으로 인해 로봇의 기대 위치와 실제 위치가 달라진 값을, 같도록 보정하는 함수이다. 다시말해, 이 메소드는 로봇의 현재 위치와 기대되는 위치가 다른 경우에만 호출되며, 현재 위치와 기대위치가 같아질때까지, 계속 호출된다.

4.4 Interaction Diagram (PlanPath)



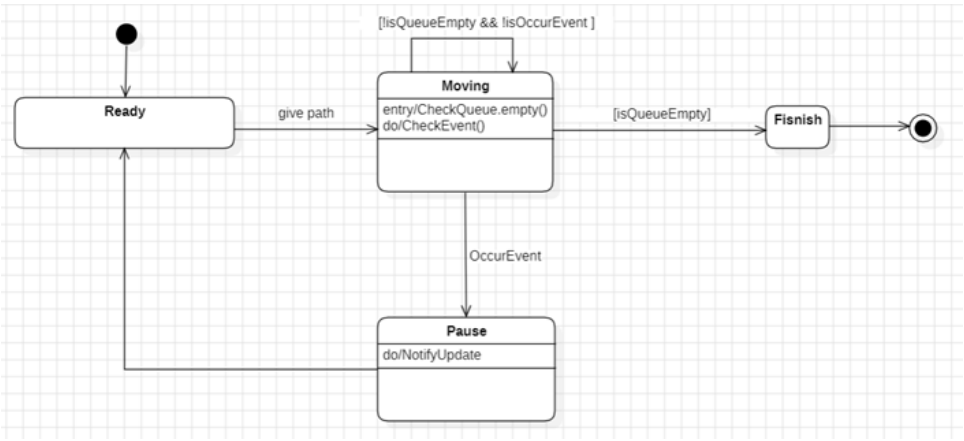
로봇의 이동 경로를 계산하기 위해서는 재난 지역 모델, 로봇의 현재 위치가 모두 필요하다. 위 교류도는 재난 지역 모델만을 받아 로봇의 이동 경로를 계산할 때 나타나는 다이어그램이다.

RequestToGeneratePath(opArea) 는 PathGenerator 의 OParea 정보를 수정한 후, PathGenerator 에게 경로 생성을 요청한다.

이때, Pathgenerator 는 로봇이 움직이는 경로를 전달하기 위해, RobotMovementInterface 의 주소값을 가지고 있으며, 이를 통해 SensorInterface 에 접근할 수 있다. 따라서, Pathgenerator 는 RobotMovementInterface 에게 로봇의 위치를 질의를 하고, RobotMovementInterface 는 SensorInterface 에게 로봇의 위치를 질의를 함으로써 Pathgenerator 에게 값을 전달한다.

이후, 전달받은 정보들을 토대로, GeneratePath 메소드가 실행되어 경로를 설정하게 된다.

5. 객체 상태도



해당 객체 상태도는 RobotMovementInterface 객체에 대한 상태도이다.

1. Ready

목적지가 없는 상태이다. 일반적인 경우 PathGenerator 에게 질의하여 목적지를 얻은 뒤 해당 상태를 빠져나간다.

2. Moving

PathGenerator 에게 질의하며 목적지를 얻고, 움직인 후 SIM 의 센서들로부터 도착한 이벤트가 있나 검사한다.

2.1 새로운 목적지를 얻었고, 도착한 이벤트가 없는 경우
다시 Moving 상태에 진입한다.

2.2 새로운 이벤트가 도착한 경우

Pause 상태로 진입한다.

2.3 새로운 목적지가 없는 경우

Finish 상태로 진입한다.

3. Pause

OperationAreaInterface 에게 재난 지역 모델의 변경을 요청하며, Ready 상태에 진입한다.

4. Finish

모든 작업이 끝난 상태.