

Rescuecore introduction

rescuecore-API

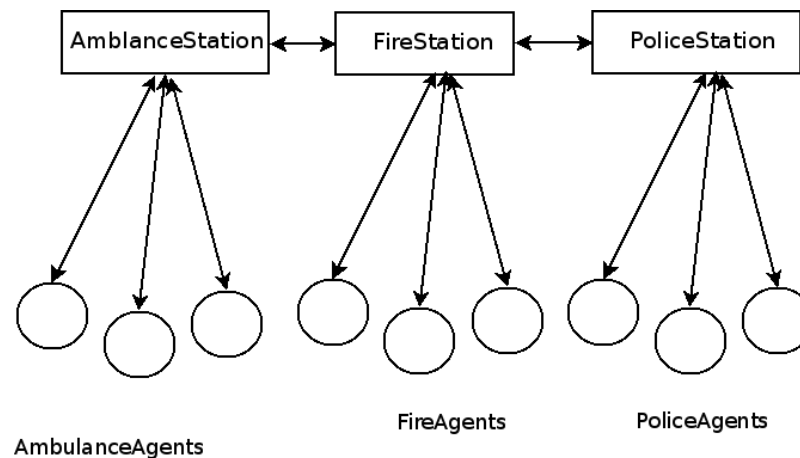
- Agent is base for all agents
- PlatoonAgent are moving agents (firebrigades, ambulances, police)
- CenterAgents are stationary (PoliceStation, etc.) – mainly communication relay
- sample/Sample... contains examples

rescuecore-API

- Agent's implement „Agent“ model
 - sense() function is called every round
 - agent should reply with sending a command to server (e.g. extinguish(42);)
- Memory is each agent's world model
 - Automatically updated from sensing
- RescueConstants defines important values, e.g.
MAX_EXTINGUISH_DISTANCE

Rescue Sim Communication

- All Communication goes through kernel
- Comm is limited
 - Agents 4 messages/cycle (256 bytes)
 - Stations 2 x <Nr. Agents>/cycle (256 bytes)



Rescue Comm Channels

- Center-/PlatoonAgents can „tell()“ messages to a channel (use 1+, not 0)
- Select channels to listen to „channel()“
- Agents have to decide to listen to a message „willListenHear()“
- Agents then „hear()“ a message

Implementation

- Use Center as blackboard
- Derive your agent from `map09.BaseExplorationAgent`
- Do NOT use `SampleExplorationAgent/TestCenter`
- For now: Use `moveTo(ExplorationTarget t)` function from `BaseExplorationAgent`, not `move(int[] path)`

Implementation Details

- Agent's world model in Memory class
- You can lookup() objects by id!
- Updated automatically
- Only includes „sense“ data
- You are responsible for additional information from com
- E.g. explorationTargets updates

Start skript

- Supply a start skript for each exercise starting the appropriate agents.
- You can use `sampleagent.sh` as a base
- Startup Parameters for `rescuecore.Launch`:
 - “<Number of agents> <class> <parameters>“
 - E.g. “3 rescuecore.map09.SampleExplorationAgent \$MAP T“
 - Number = 0 tries to connect as many agents as possible