

CITY DYNAMICS

A MULTI-AGENTS SYSTEM

FINAL REPORT PRESENTATION

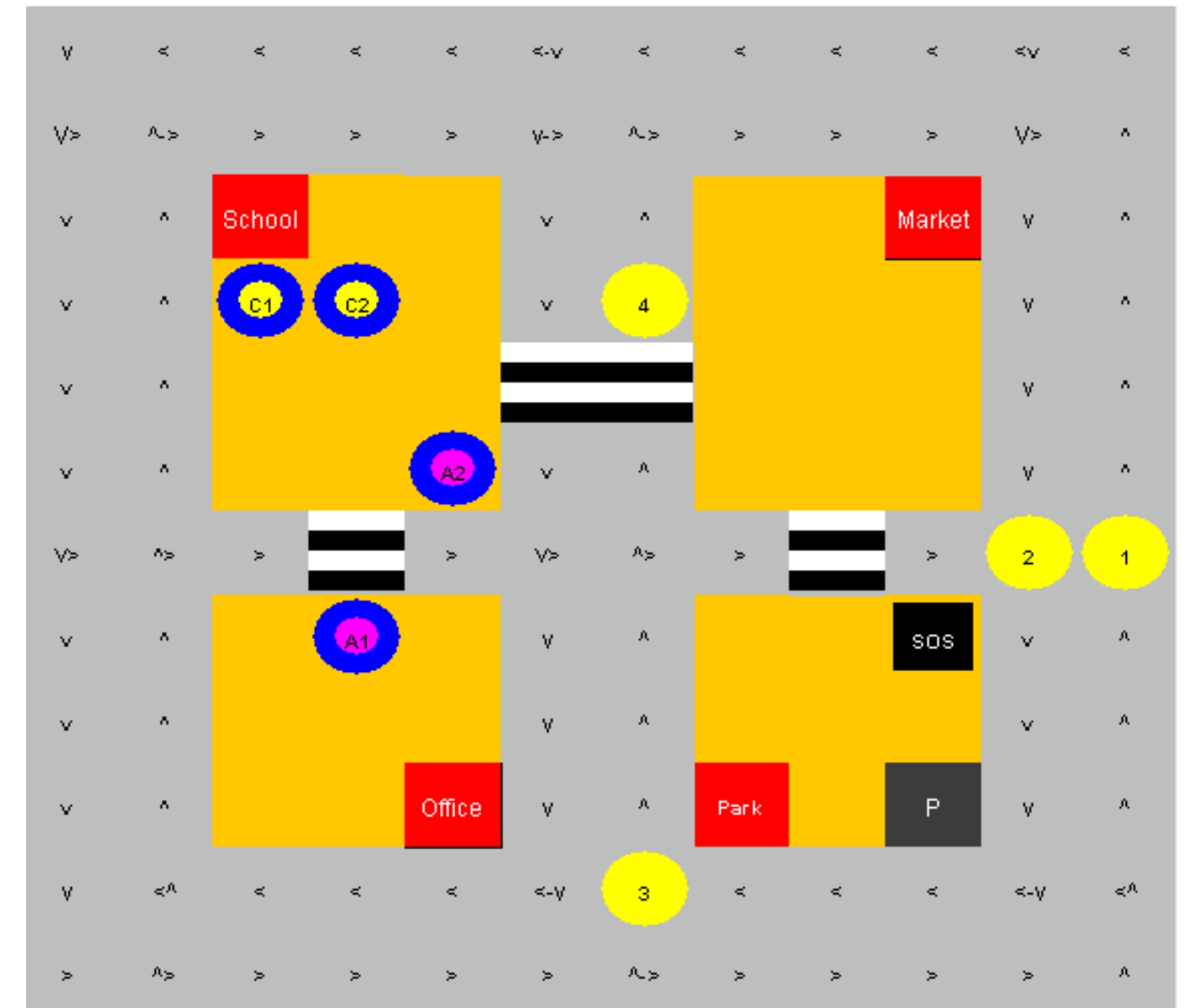
ALICE TURRINI

MAUD RAVANAS-DESHAYS

Idea of the project

*“Simulate the **traffic flows** of a city, this is multi-agent system with **cars** and **pedestrians** as agents.*

*They are **autonomous and intelligent**, relying on their perception of the environment to achieve specific goals.”*



Requirements

Goals: Environment



- **Buildings:**

- they are for pedestrian only
- **Special buildings:** a *supermarket*, a *school*, a *park* and an *office*
- Zebra crossing connects them

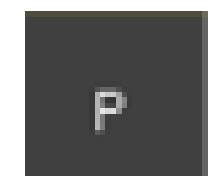


- **Streets:**

- each lane has one or two direction(s) among: *up*, *down*, *right*, *left*
- crossroads: bidirectional blocks where one of the two direction possibly is assigned with precedence



- **Zebra crossings:** street blocks but for pedestrian walking

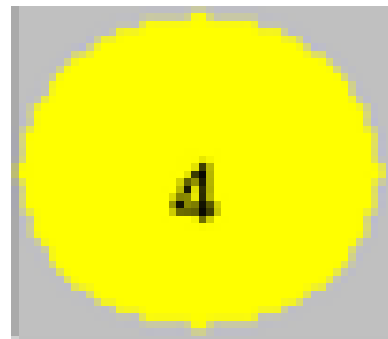


- **Garage:** parking spot of the helicopter

Goals: Vehicles agents

- **Cars:**

- **Drive** in streets following *traffic regulation* and **adapt** to the *current traffic state*
- **Can't share position**, so change direction but not going back.
- Randomly **break down** and stop moving

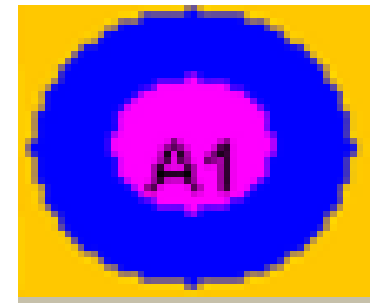


- **Helicopter:**

- **Flies**, so no need to follow *traffic regulation*
- **Fixes** the broken cars; once repaired, it goes back to its parking spot

Goals: Pedestrian agents

- **Pedestrians:**
 - **Head towards** *specific buildings*:
 - *adults*: they go to the office, and then to do the grocery shop at the supermarket
 - *child*: they go to the school, and then to play at the park.

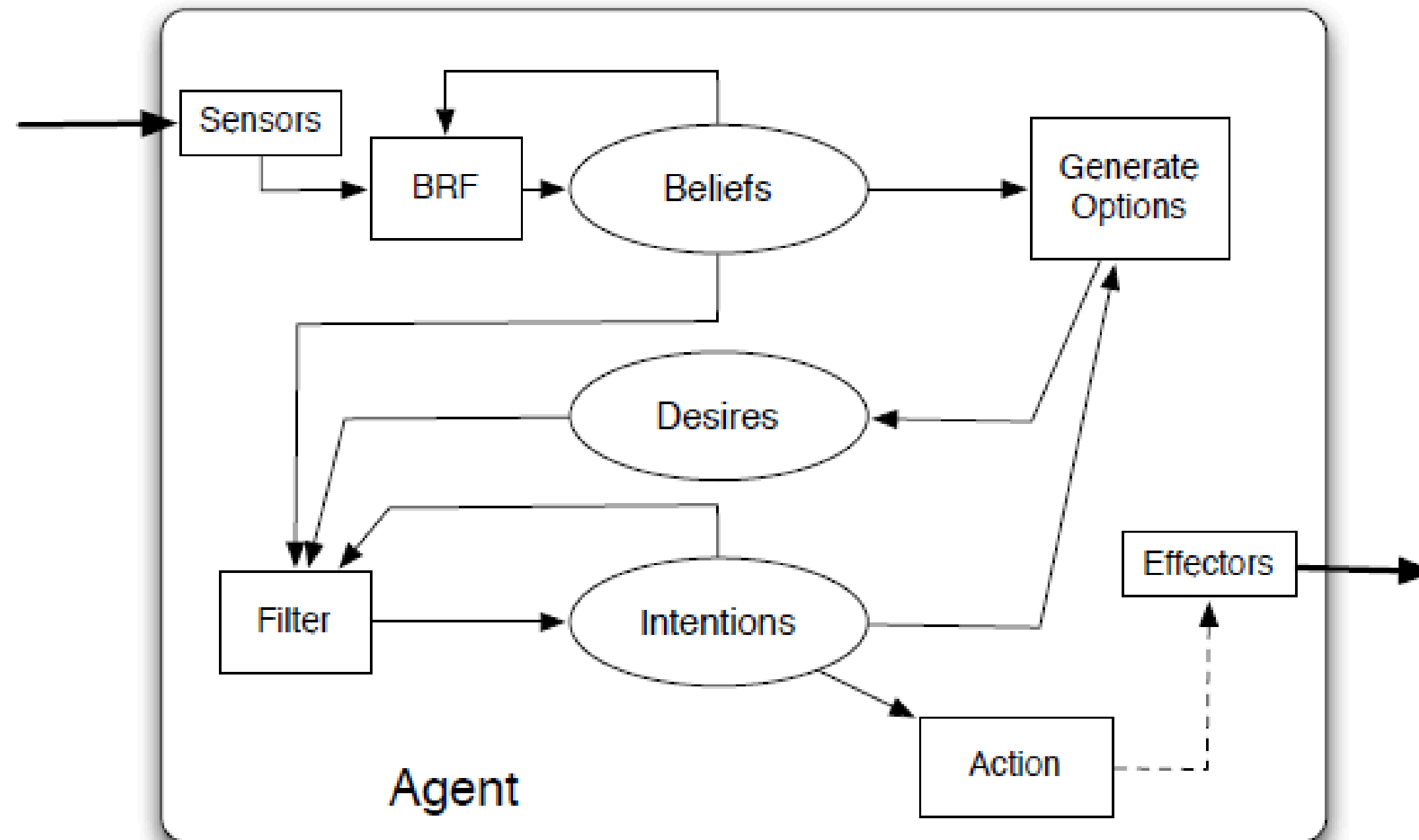


- **Walk** in buildings and on zebra crossings
- Can share the same location

Design of the

MAS

BDI Architecture



Goal-directed agents acting both:

- **proactively**
- **reactively**

to fulfill their purpose.

Technology used: JaCaMo

JaCaMo is a **development platform**.

JA

Jason: agent programming language java-based language extending ASL in respect of BDI architecture.

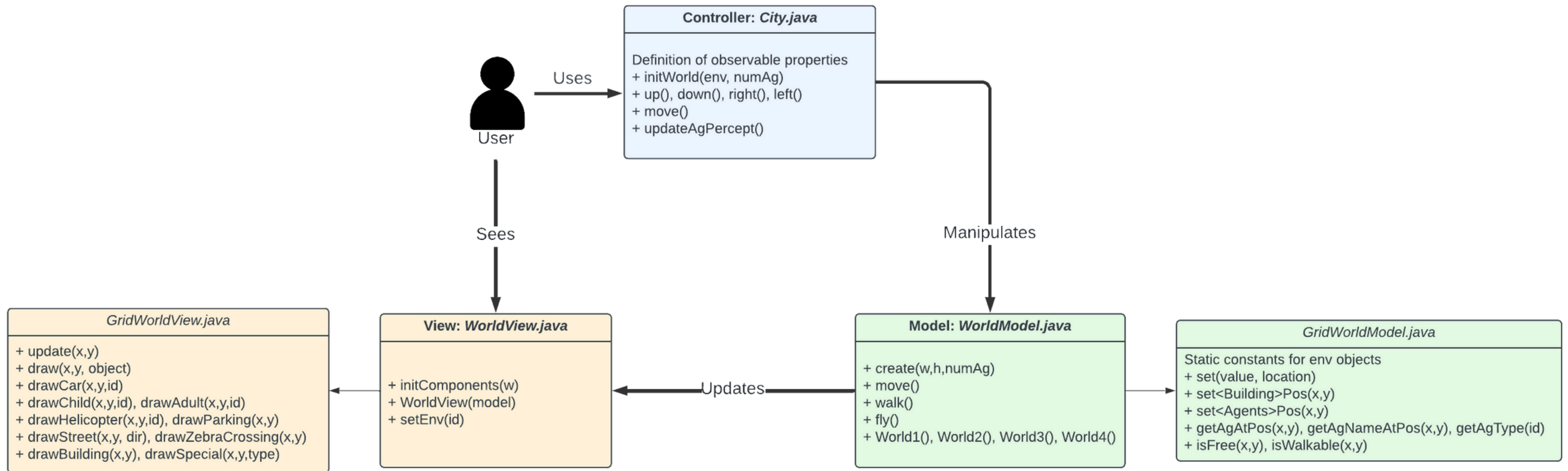
CA

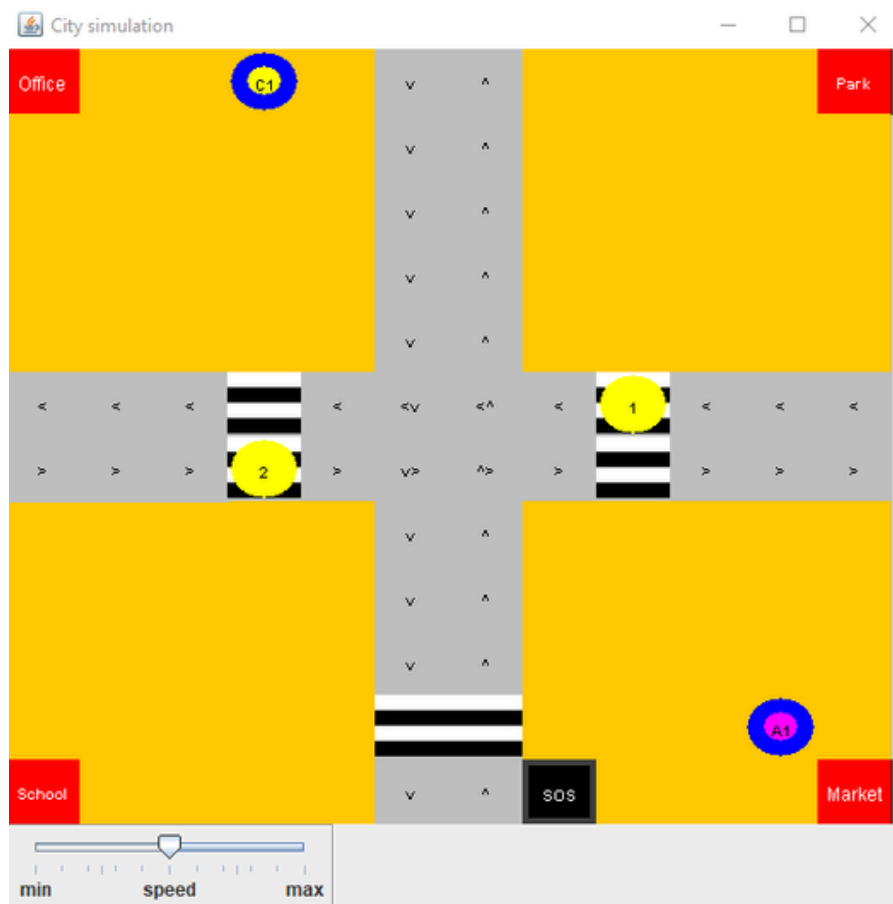
Cartago: framework for creating, managing, and interacting with artifacts in a shared custom environment

Mo

Moise: organizational model (not used in our case)

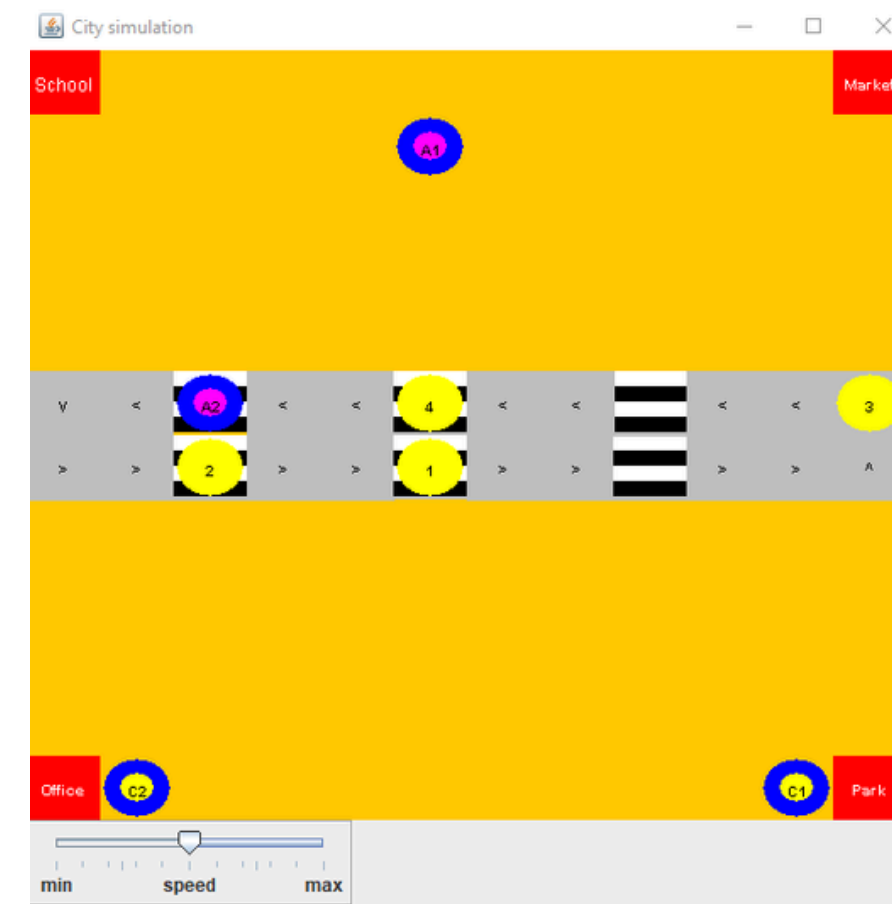
MVC pattern



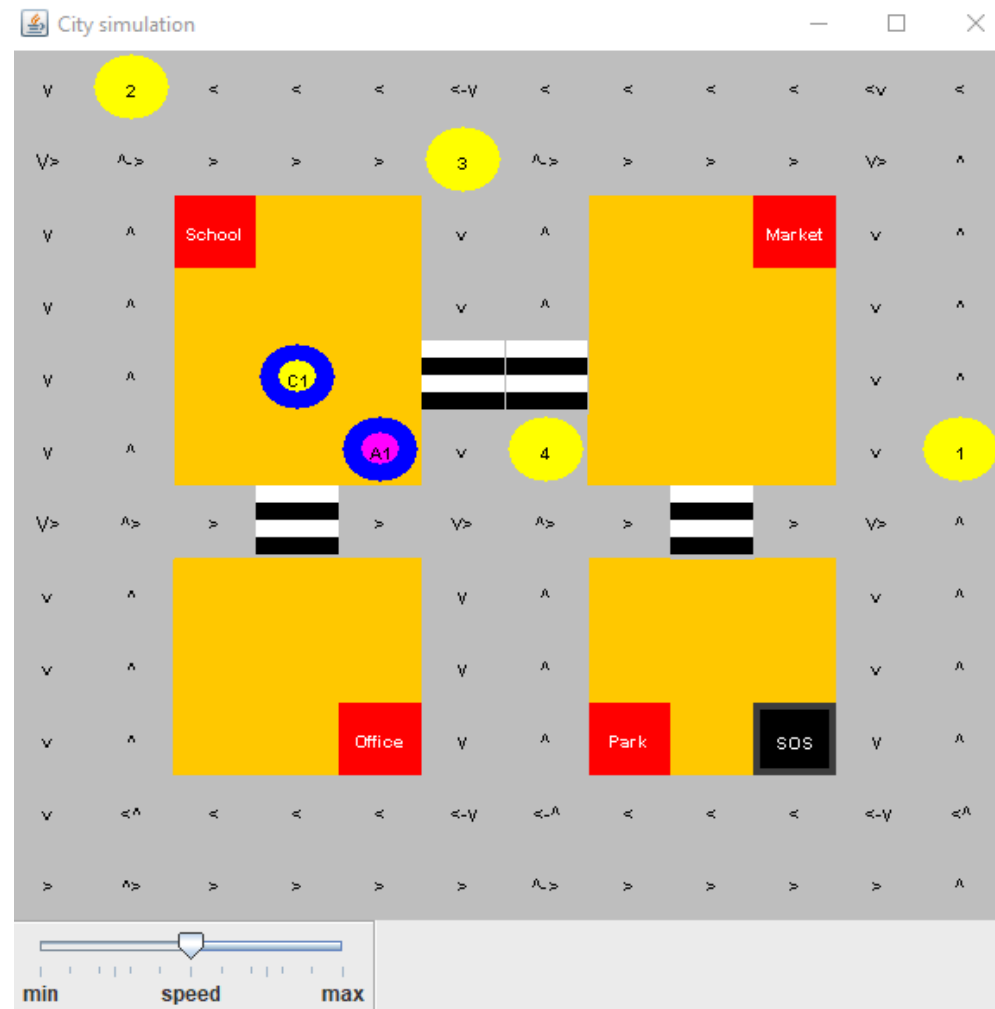


1

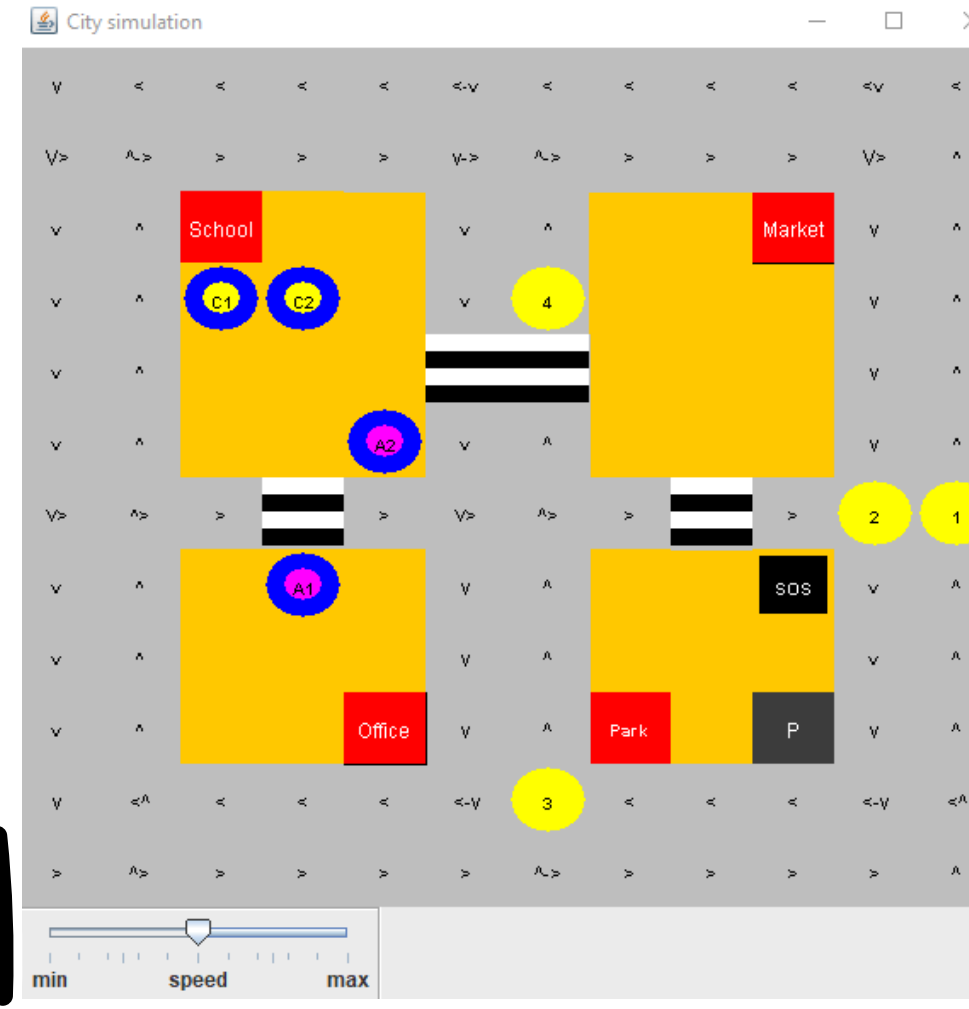
4 Different Worlds



2



3



4

Design of Agents

Cars, Pedestrians, Helicopter

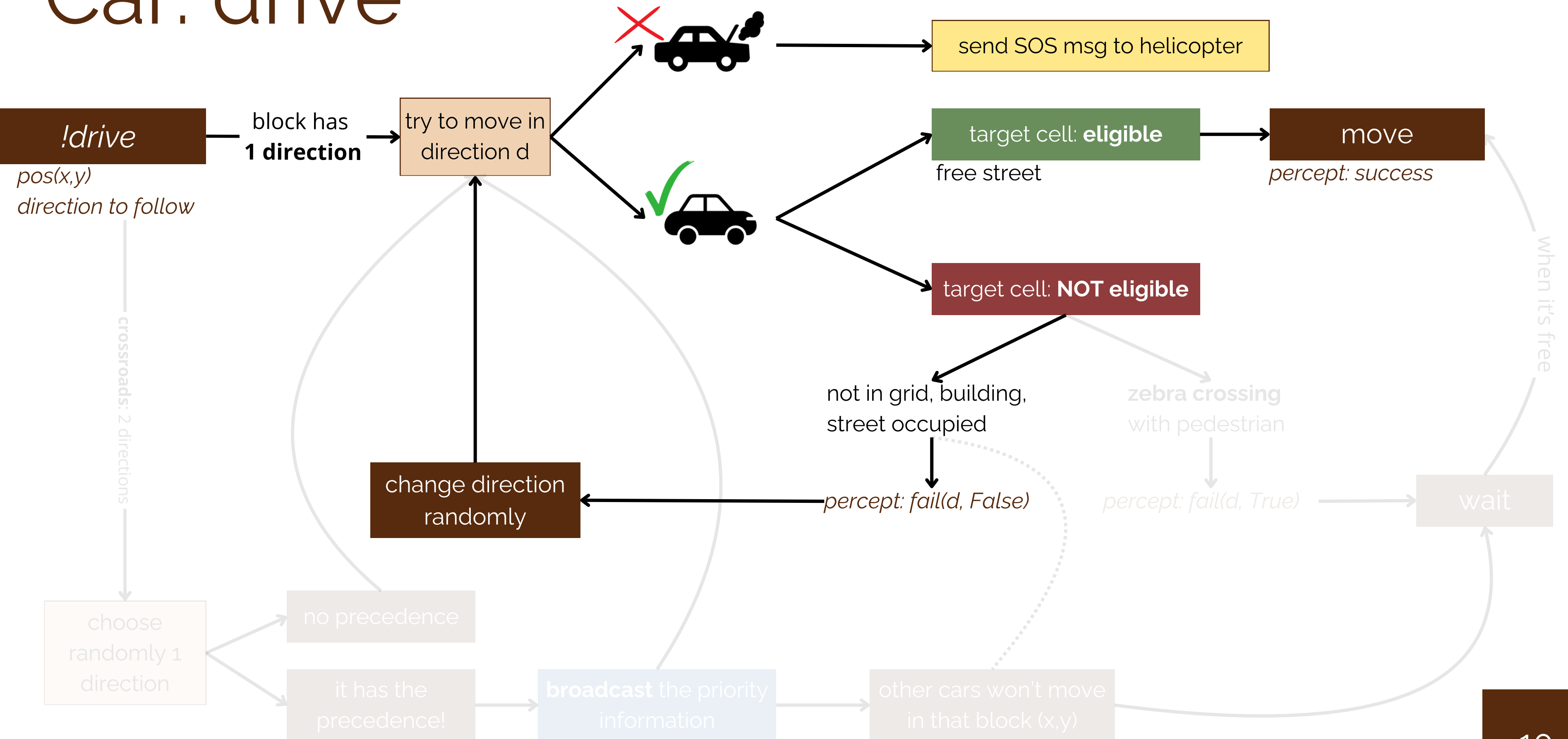
Coordination

- **Cars on lanes**
 - avoid congestion
- **Cars at zebra-crossings**
 - avoid crashes
- **Cars at crossroads**
 - respect precedence

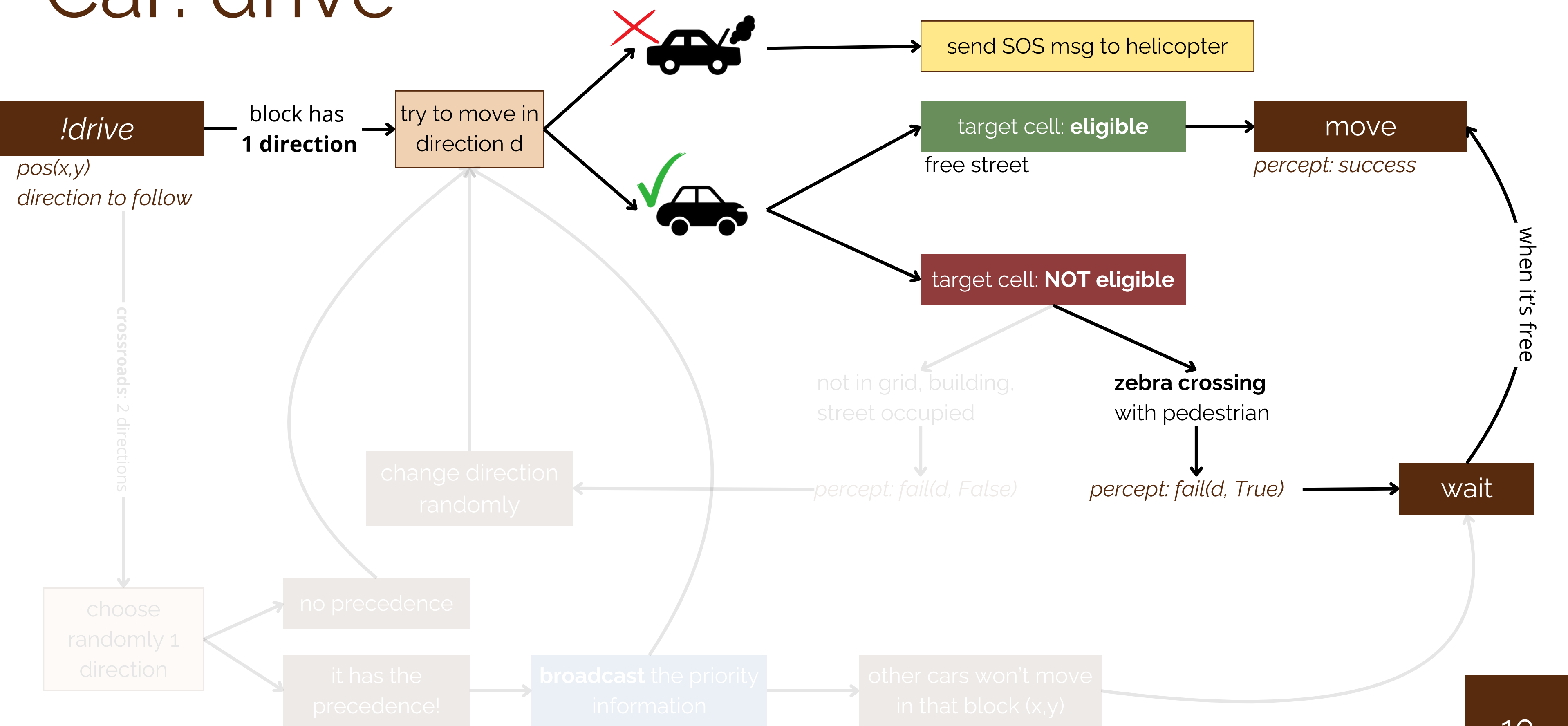
Communication

- **Helicopter fixing the cars**
 - avoid blocking streets because of individual technical issues
- **Pedestrian being welcoming**
 - greet each other

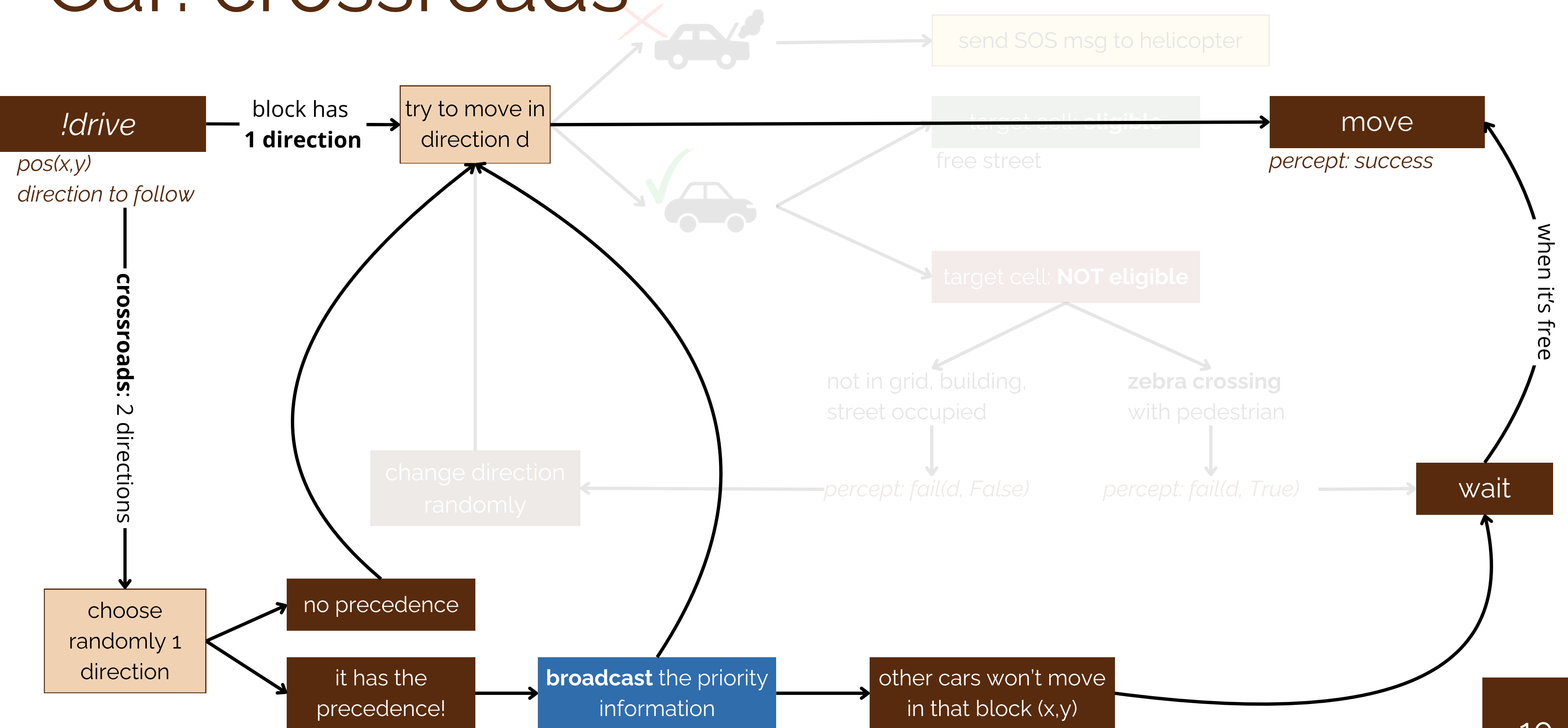
Car: drive



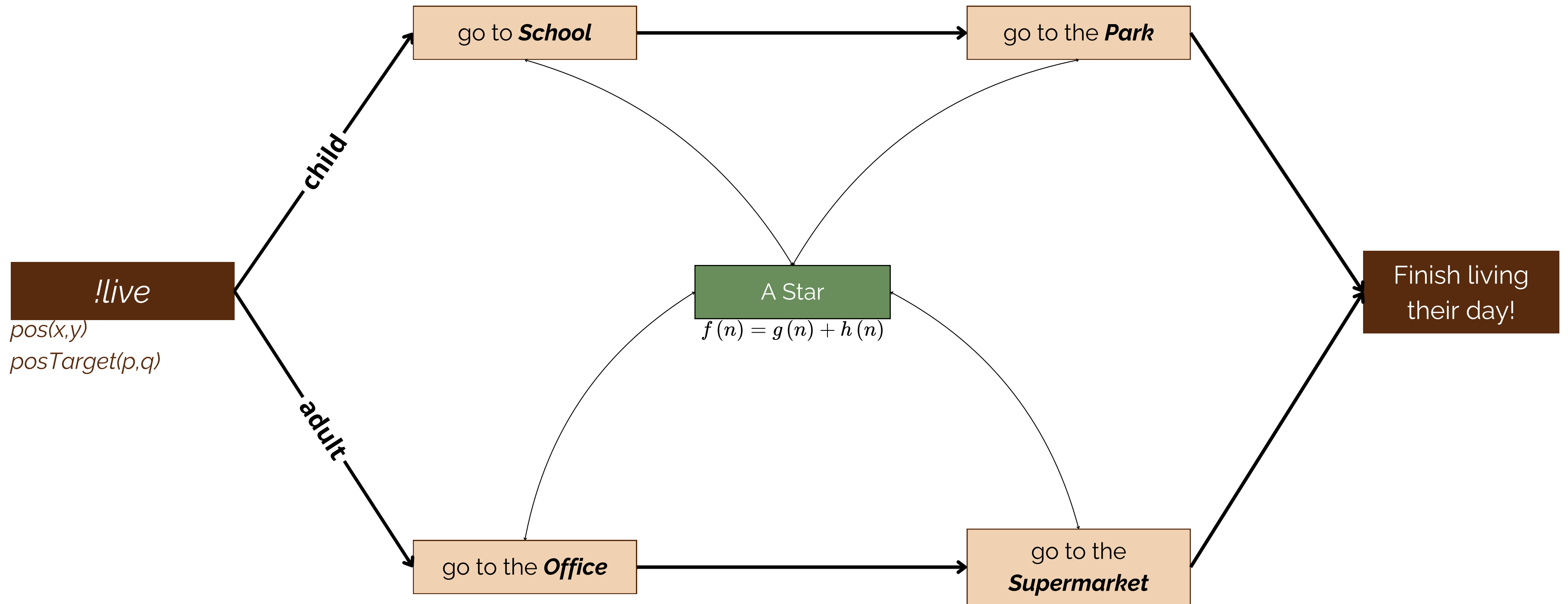
Car: drive



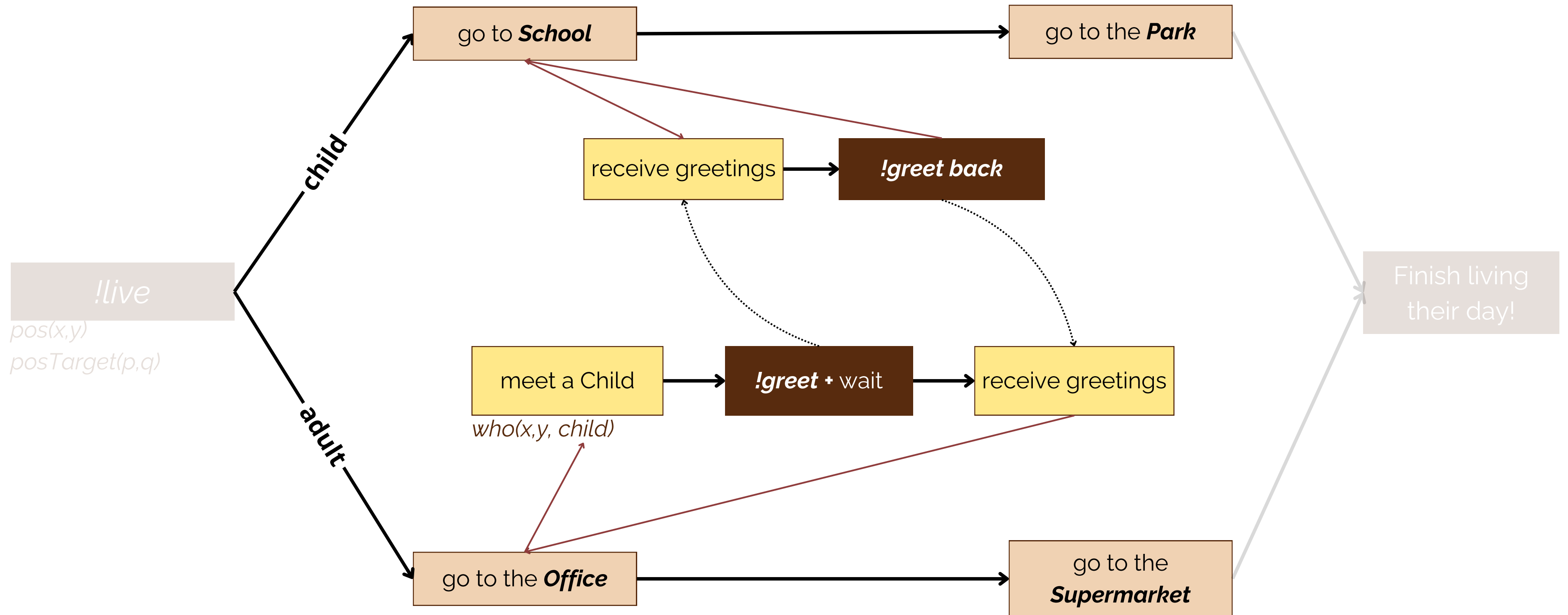
Car: crossroads



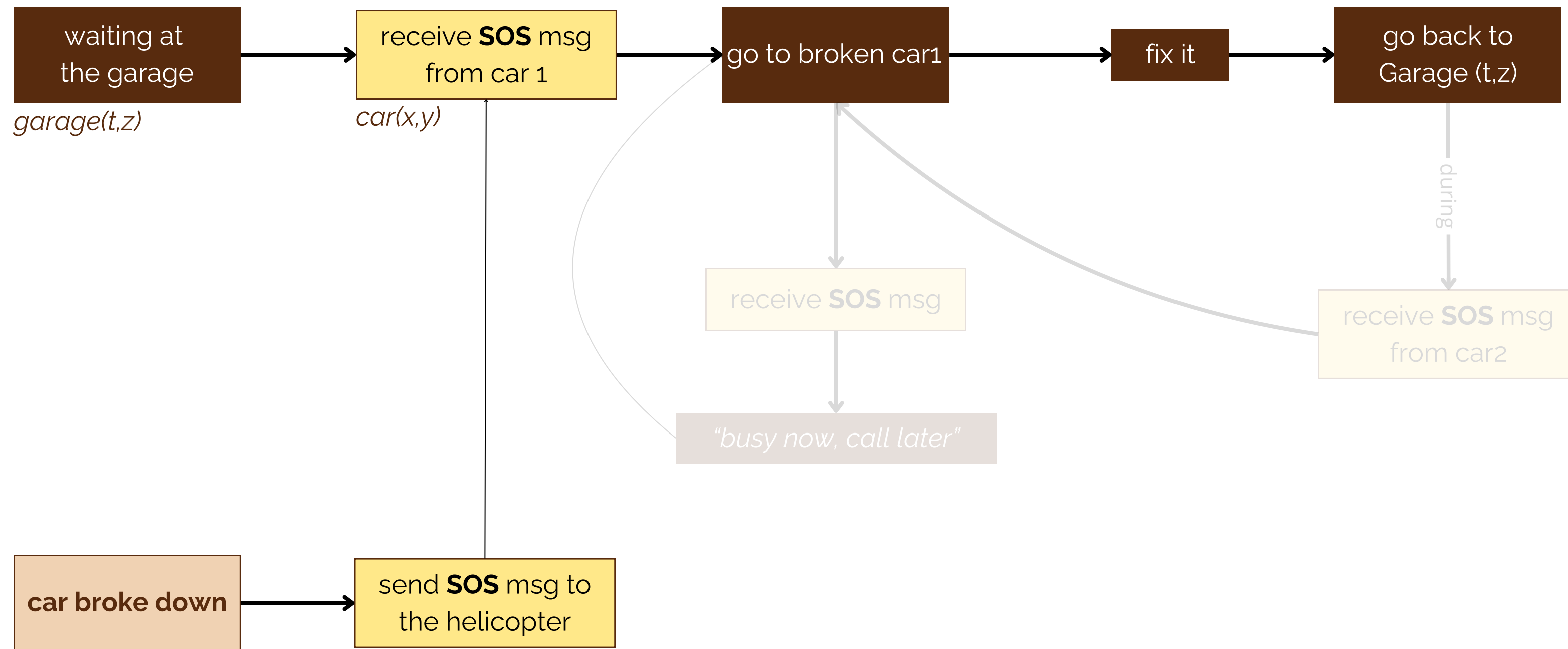
Pedestrian: living a day



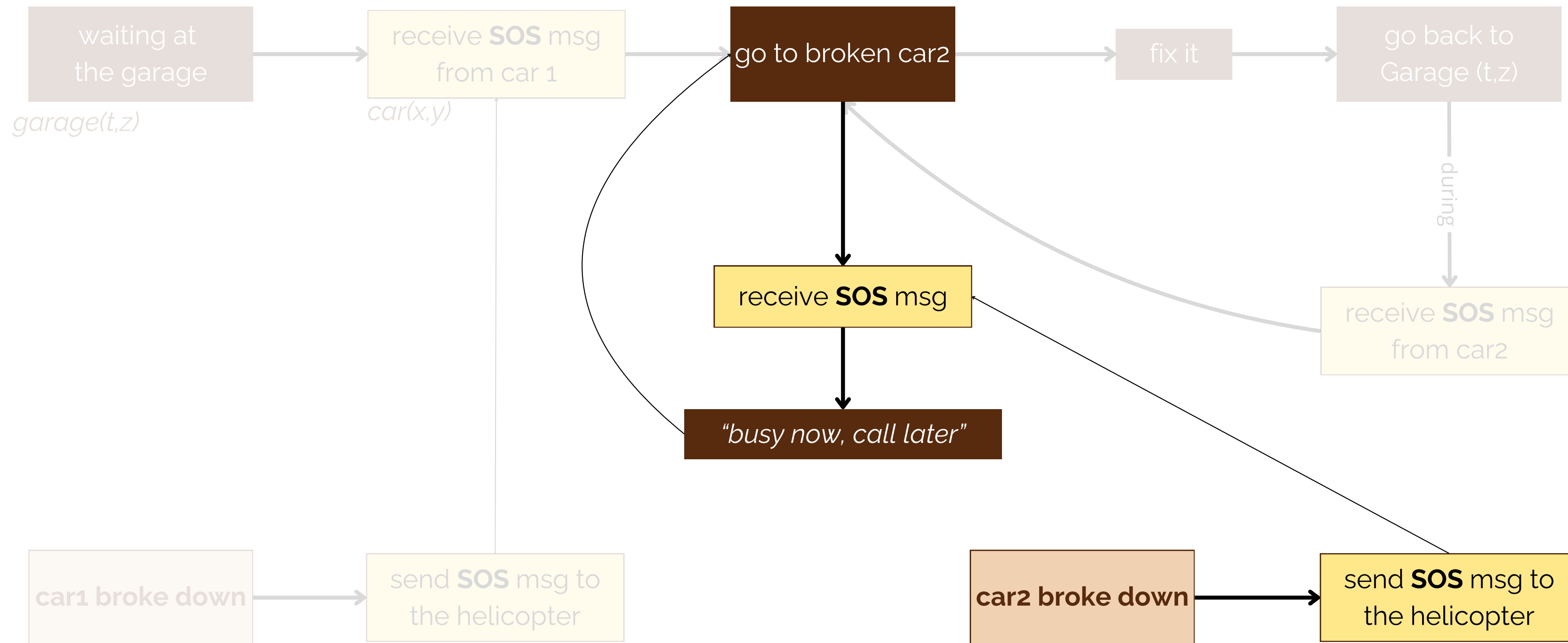
Pedestrian: greetings



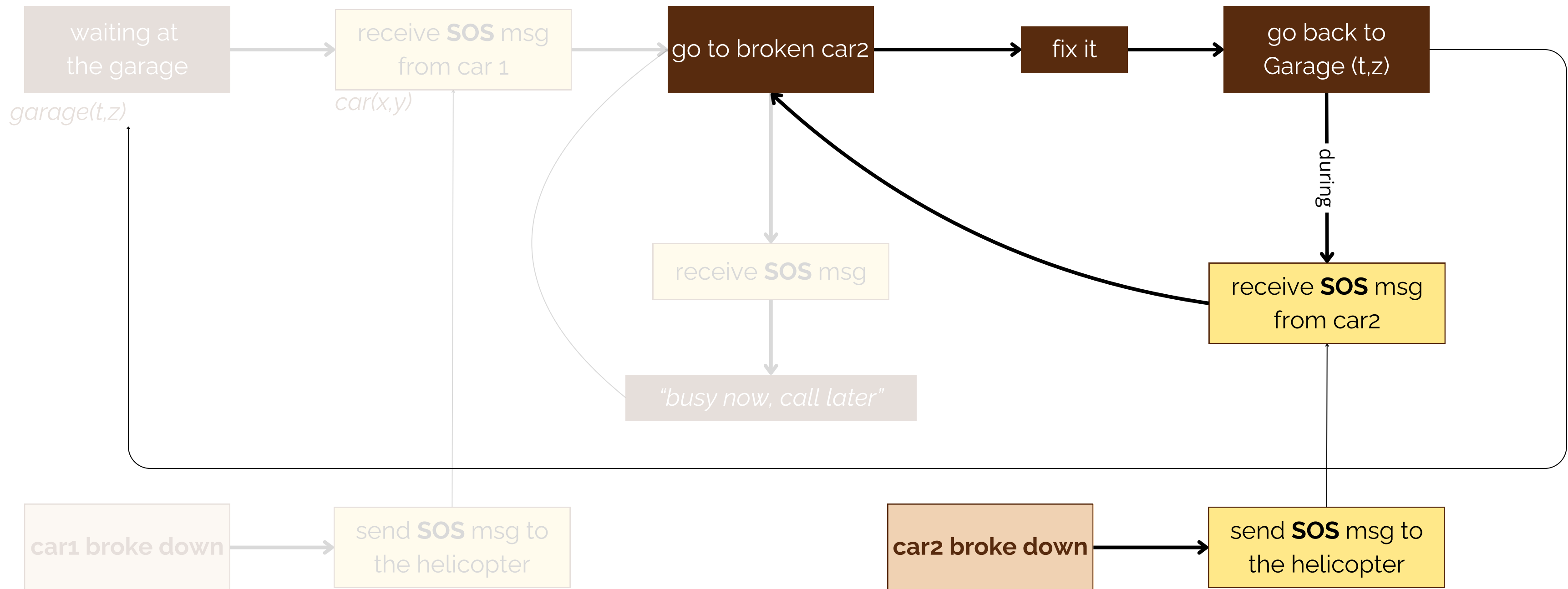
Helicopter: fix a car



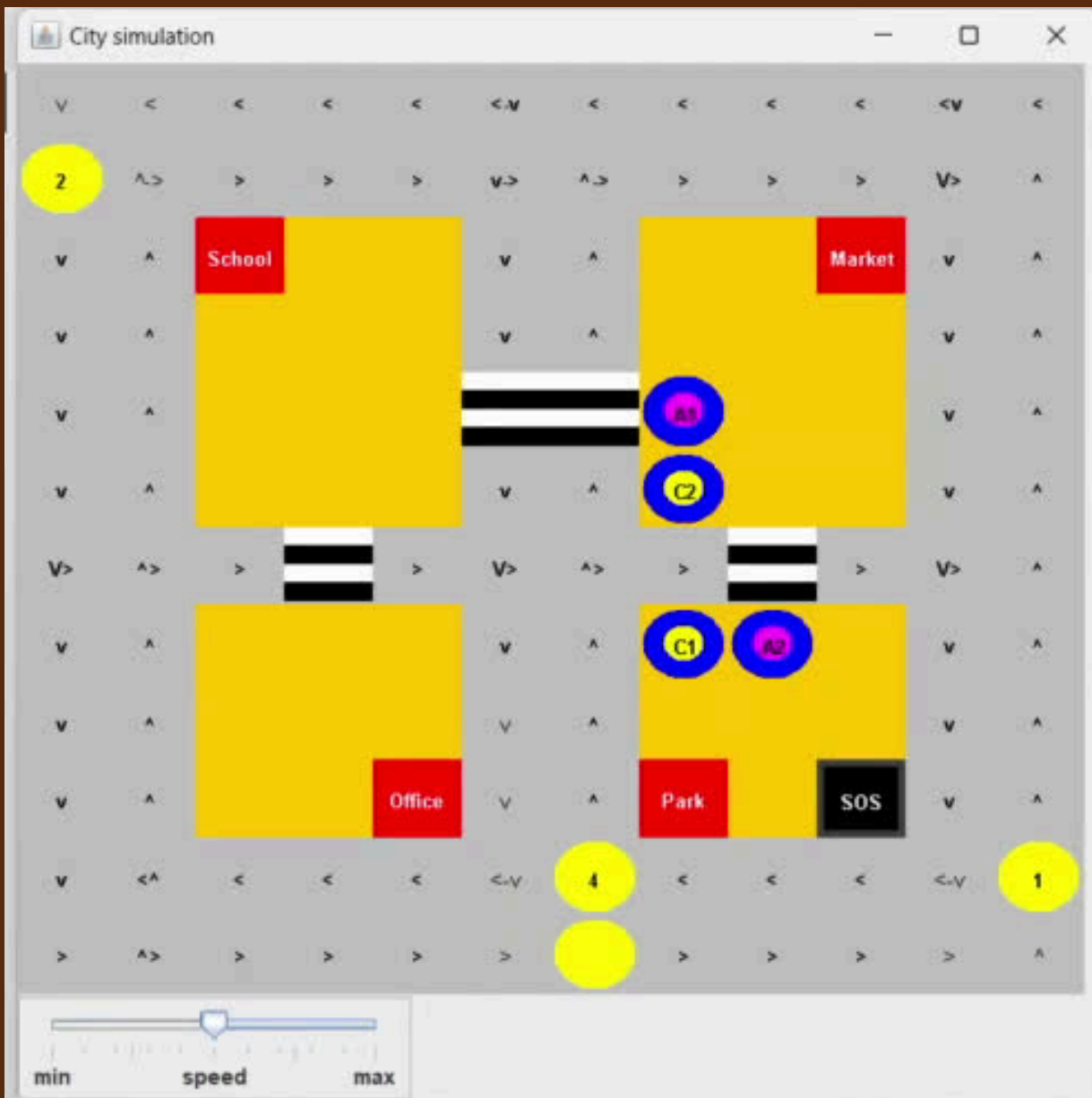
Helicopter: fix a car another car is calling



Helicopter: fix a car another car is calling



Demo



mas-project-turrinideshays2324 (Workspace)

middle (and zebra crossing, different buildings and helicopter)

ter see their communication

Task :run

*** The jason.jar from classpath is different than jason.jar from configuration, consider to delete the configuration (file ~/.jason/user.properties) or 'uns
Classpath is
C:\Users\alice\gradle\caches\modules-2\files-2.1\io.github.jason-lang\jason-interpretor\3.2.1\36ce5ab2bfbc80c6935e2d217cb88b66cbd61de\jason-interpretor-
Config is
/C:/Users/alice/.gradle/caches/modules-2/files-2.1/io.github.jason-lang/jason-interpretor/3.2.1/36ce5ab2bfbc80c6935e2d217cb88b66cbd61de/jason-interpretor-
Using the jason.jar from classpath

<===== 75% EXECUTING [3s]

> :run

OUTLINE
TIMELINE
JAVA PROJECTS

master 2 0 W 0 30 Java: Warning

Ln 82, Col 49 Spaces: 4 UTF-

MAS Console - city_simula... City simulation

Questions' time

We thank you for your attention!