Development Framework For Search and Rescue Agents

Topics

- Introduction
- Domain
- Existing frameworks analysis
- The CLRF framework
- Building an agent
- Conclusions

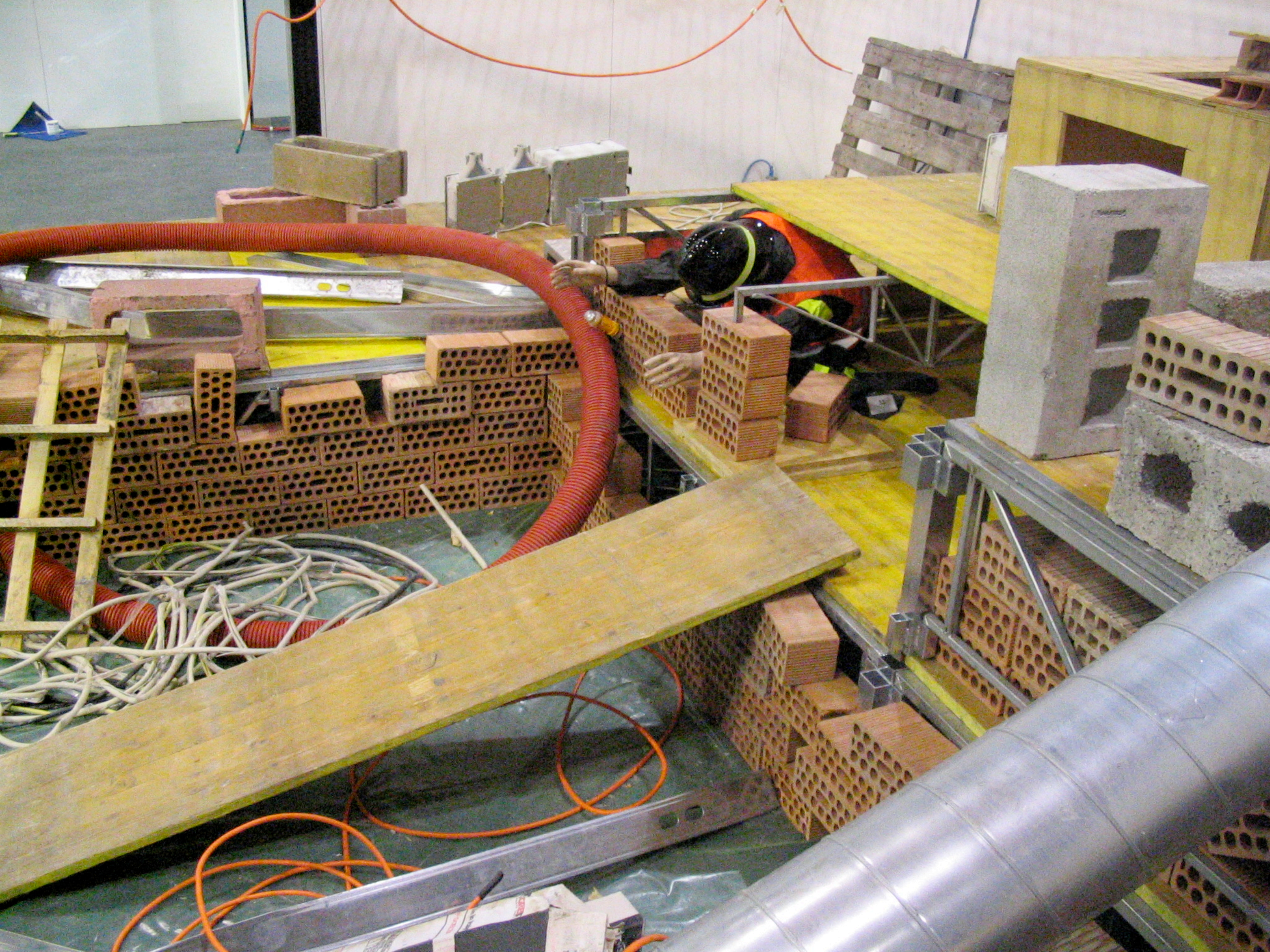
Topics

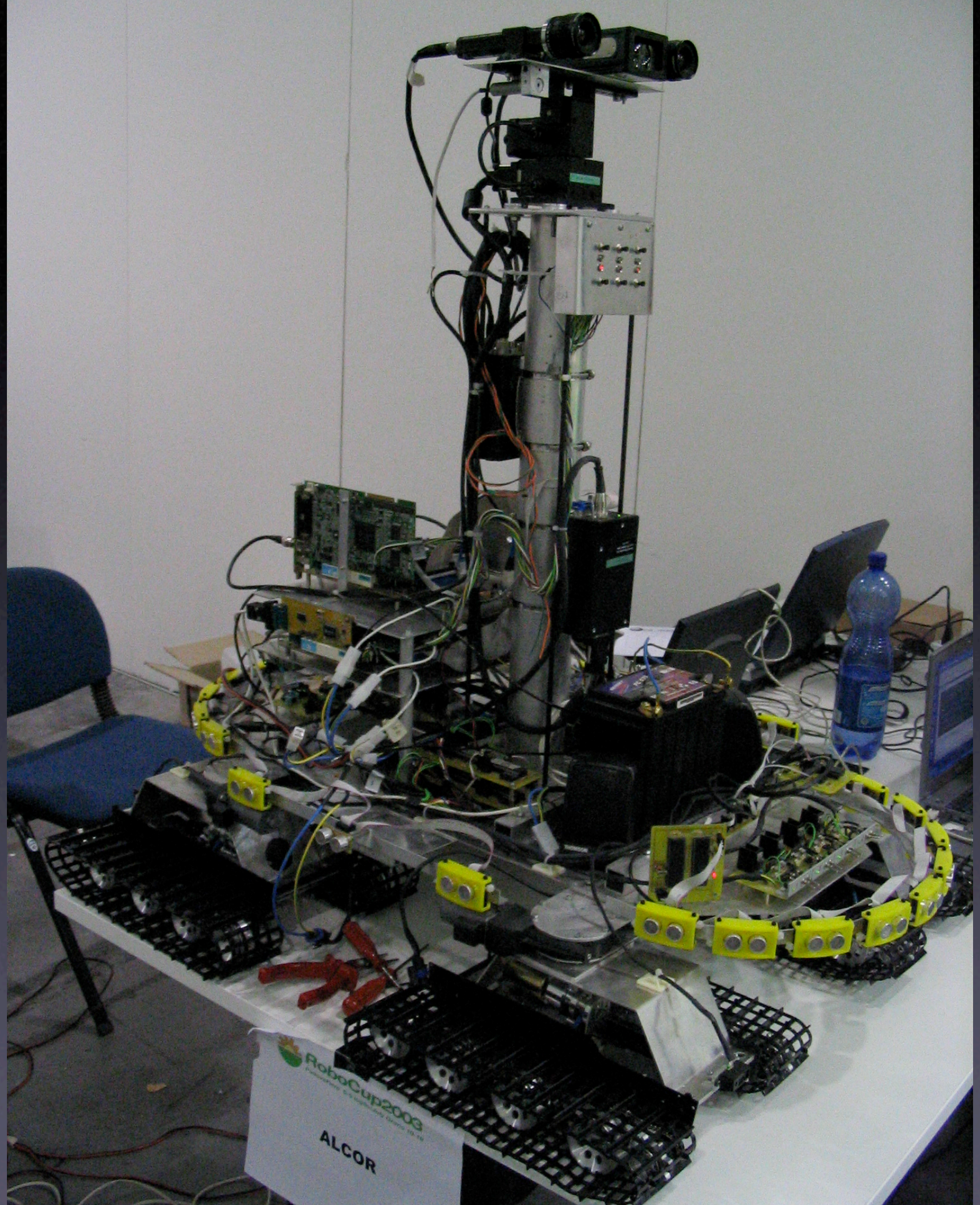
- Introduction
- Domain
- Existing frameworks analysis
- The CLRF framework
- Building an agent
- Conclusions

Introdução

- RoboCup – worldwide project aiming the development of new technology in the robotics and AI fields.
- RoboCup Rescue – RoboCup project dedicated to the construction of rescue agents.


SONY
Alitalia
RoboCup Rescue Robot League
sgi
ZENRIN
ZENRIN
sgi



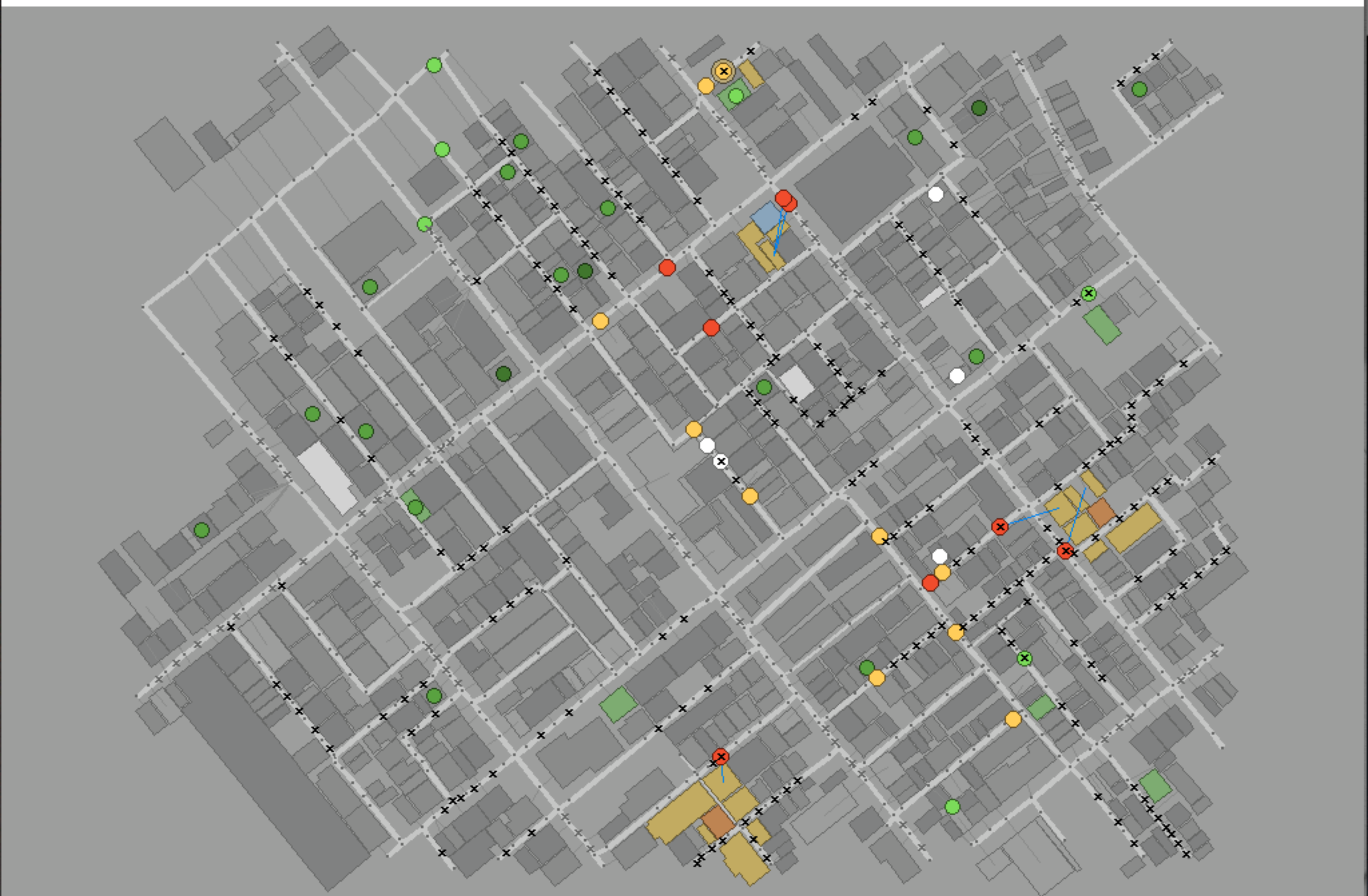

RoboCup2003
ALCOR

Introduction

- Rescue Simulation – group of virtual agents that act on a simulated world.
- Scenario: a city, immediately after the occurrence of an earthquake.
- Goal: save as many human lives as possible.

X11 Applications Edit Window Help (99%) 11:37

Time: 30 Score: 96.530177



Interval Animation

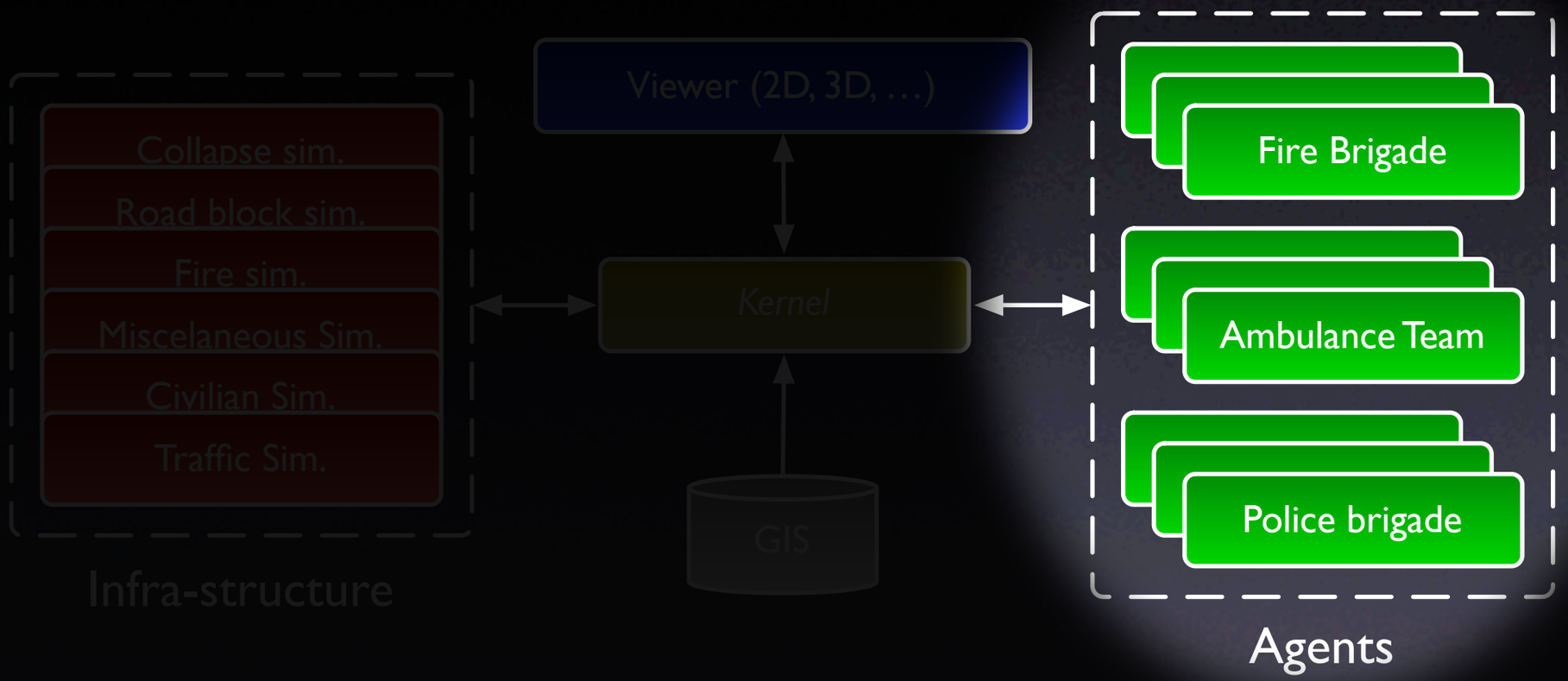
Introduction

- The existing rescue agent building frameworks have many limitations.
- There was the need to create a new one.

Topics

- Introduction
- Domain
- Existing frameworks analysis
- The CLRF framework
- Building an agent
- Conclusions

Architecture



Agents

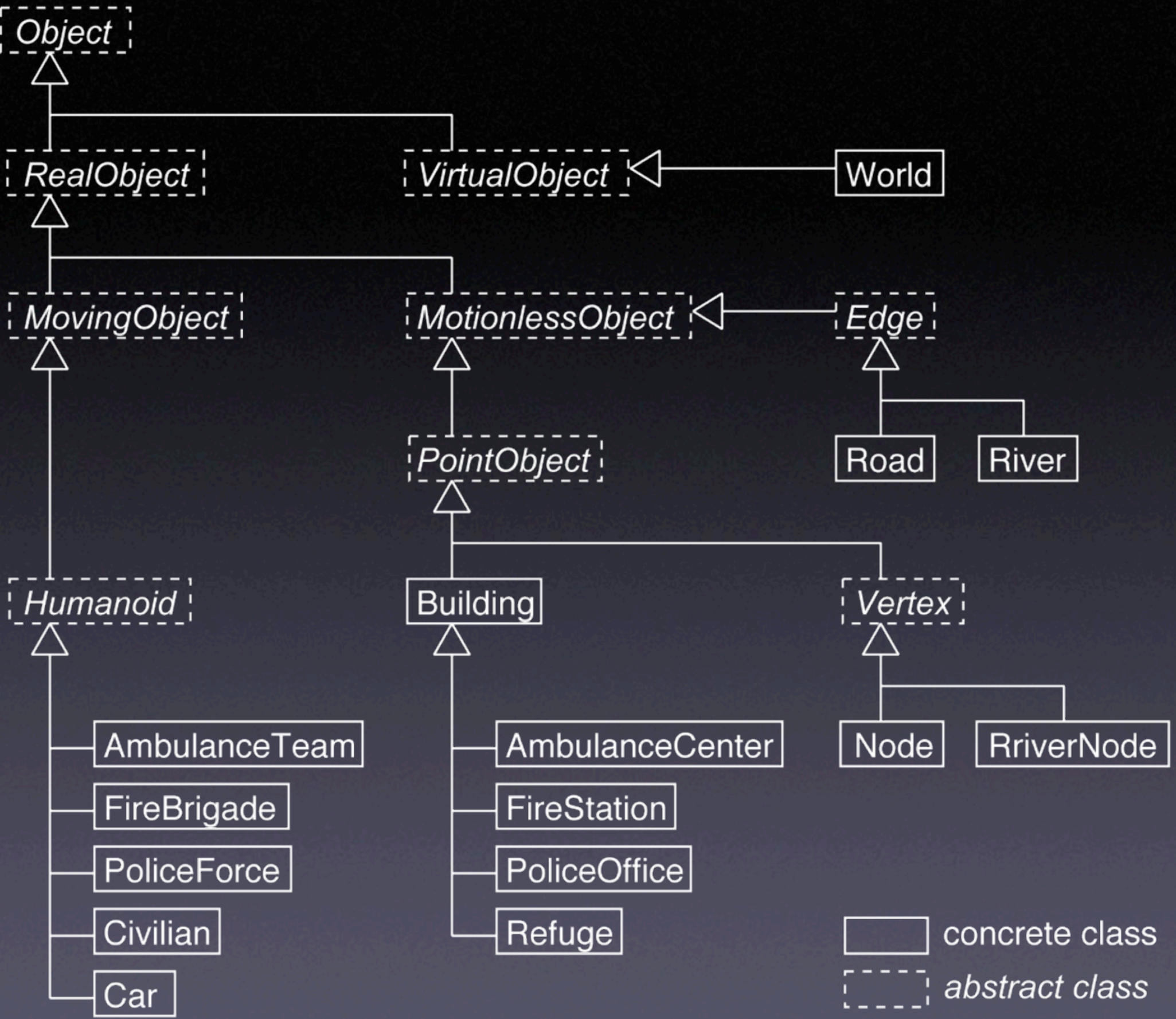
- There are six types of agents:
  - Police forces, fire brigades, ambulance teams;
  - Police offices, fire stations, ambulance control centers.

Communications

- The communication between the components is carried out using UDP sockets.
- Each message has an header, one or more objects and properties.

World representation

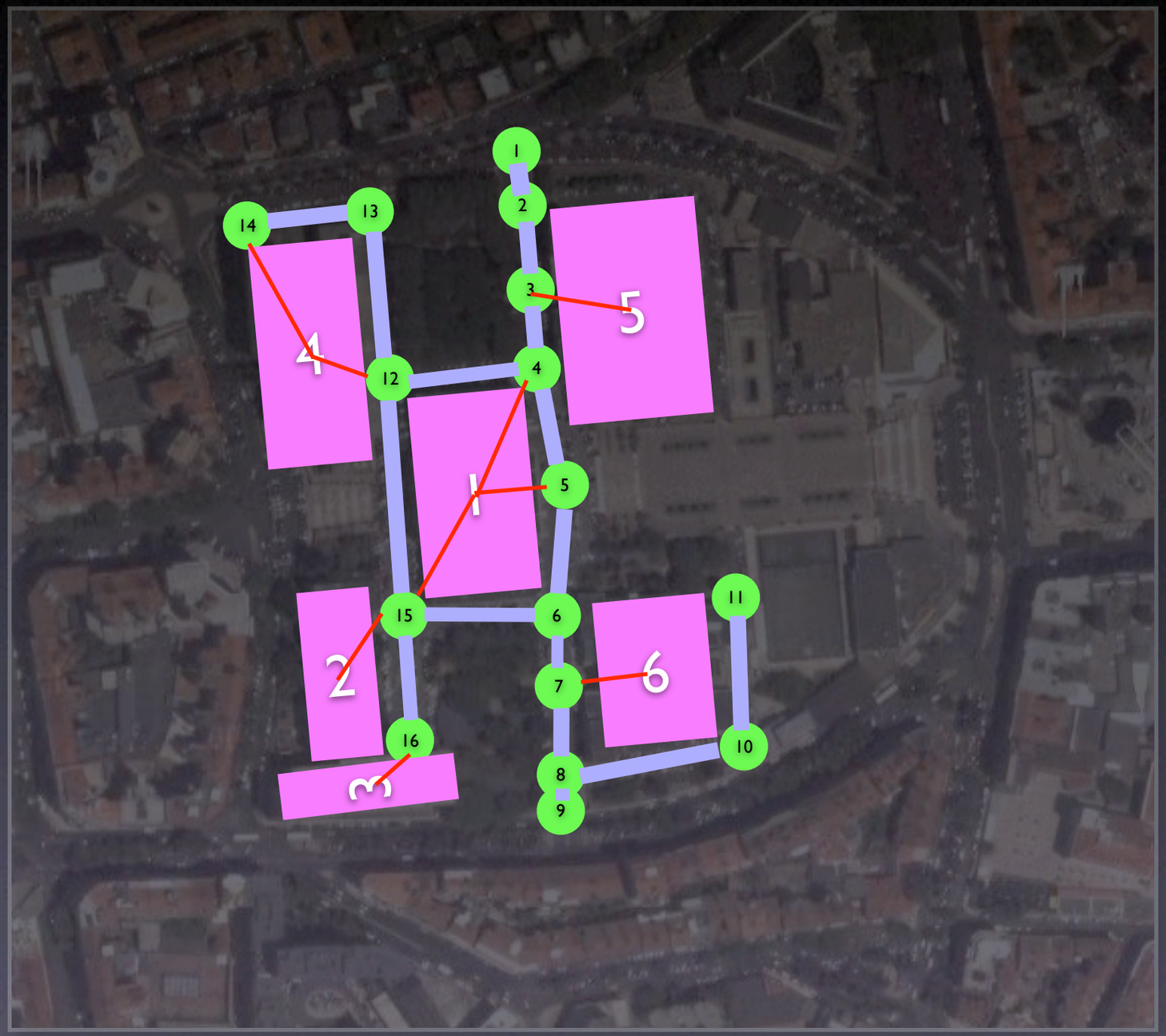
- Class hierarchy.
- Each object has a set of static or dynamic properties.
- Each object is identified by a unique number.
    - That number is different on each agent!


Object
RealObject
VirtualObject
World
MovingObject
MotionlessObject
Edge
PointObject
Road
River
Humanoid
Building
Vertex
AmbulanceTeam
FireBrigade
PoliceForce
Civilian
Car
AmbulanceCenter
FireStation
PoliceOffice
Refuge
Node
RriverNode
concrete class
abstract class

World representation

Fotografia © Google Maps

World representation



Topics

- Introduction
- Domain
- Existing frameworks analysis
- The CLRF framework
- Building an agent
- Conclusions

Existing frameworks

- Two existing frameworks:
  - yabapi *(oficial)*
  - RescueCore

Threads

- An autonomous agent needs to perform several tasks concurrently:
  - Decision, action planning, path planning, handling inter-agent communications, etc…

Threads

- Several tasks at once.
- Data change and sharing between the tasks (data corruption, concurrent access, data coherence, etc)…

Threads

- Neither tool support a threaded model.

World representation

- Most intelligent agents need as internal representation of the most important characteristics of the outside world.

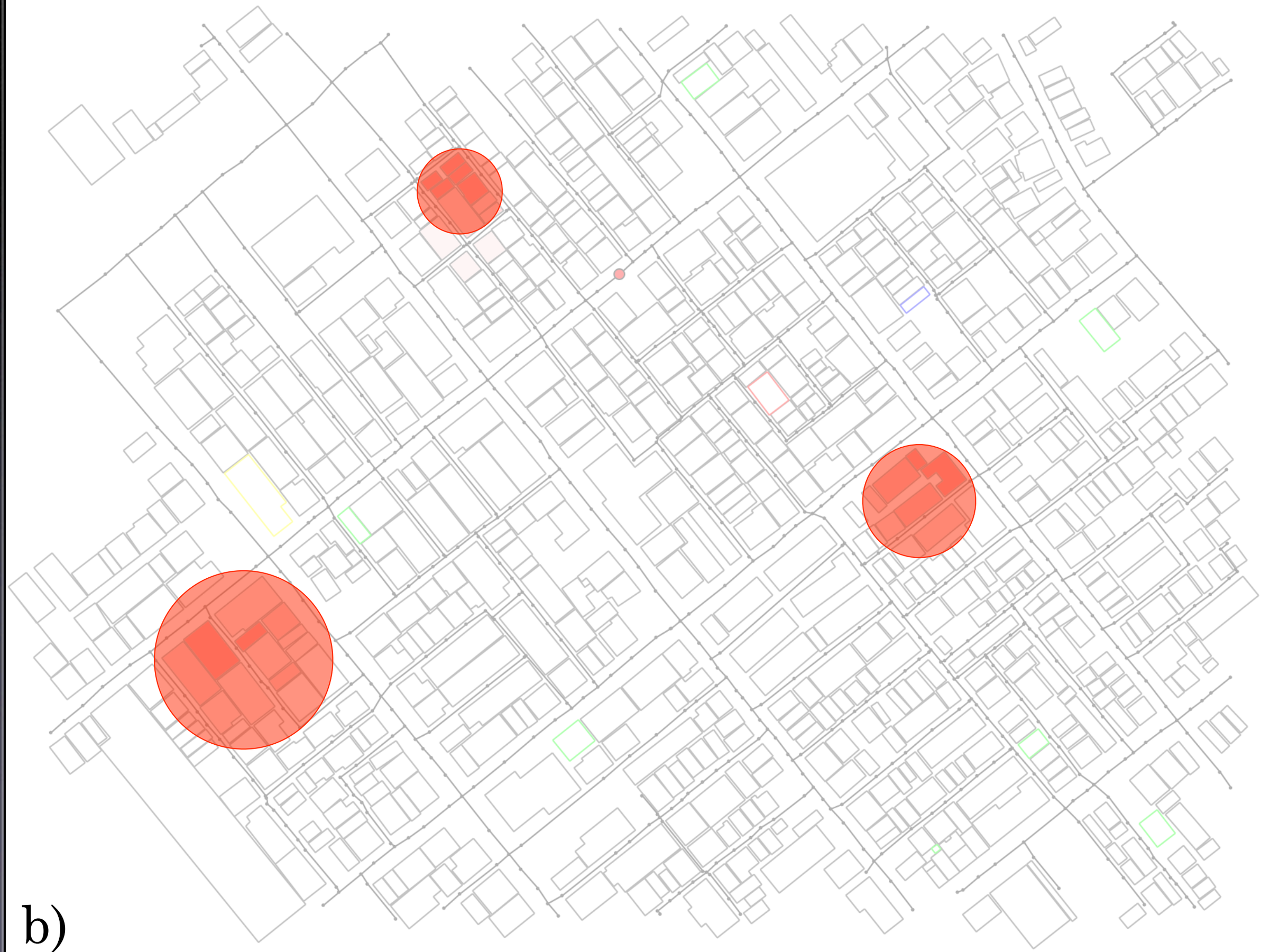
World representation

- In the RoboCup Rescue domain:
  - It's necessary to handle buildings, roads, nodes, other agents, …
  - Too much information, or data too much detailed.

World representation

- Example:
  - A fire brigade, if not extinguishing a fire and having enough water, should move itself towards *the biggest existing fire*.


a)



b)

World representation

- It's useful to manage several world representations at the same time, each one with different granularity level and focused on different world aspects.
- None of the frameworks offers built-in support for more than one world representation.

Miscellaneous

- Graphic world representation:
  - yabapi – No.
  - RescueCore – Yes, several layers.

Miscellaneous

- Logging support:
  - yabapi – No.
  - RescueCore – Yes (file).

Miscellaneous

- Documentation:
  - yapabi – No independent docs. There is some documentation that comes together with the RSS documentation, but it's not very clear and doesn't have conceptual descriptions.
  - RescueCore – Only very incomplete *JavaDoc*.

Miscellaneous

- Implementation:
  - Both frameworks are Java-based. Java is not an appropriate programming language for this environment.
  - Source code contains many errors and inelegant solutions.

Analysis

- Both frameworks are very limited and don't offer essential features that should be present on a RoboCup Rescue agent construction framework.

Topics

- Introduction
- Domain
- Existing frameworks analysis
- The CLRF framework
- Building an agent
- Conclusions

The CLRF

- Thread-based structure;
- Modular and extendable world representation;
- Graphic world representation;
- Miscellaneous tools;
- Documentation.

LISP

Threads

- Threads is good!
  - Several tasks running at the same time, automatic load balancing on machines with more than one CPU.
- Threads is bad!
  - Several threads accessing and modifying data at the same time, *race conditions*, etc.

Threads

- There is the need for something that gives the user the good stuff threads have, but that handles the bad stuff in the back.

Micro-agent

Micro-agent

- A micro-agent is an intelligent thread.
- It includes initialization, run and destruction methods.
- Queue for receiving inter-micro-agent messages.

Micro-agent

- Each micro-agent is responsible for managing and encapsulating the data it maintains.
- The access to a micro-agent's data must be handled by it, in a thread-safe way.

Micro-agent

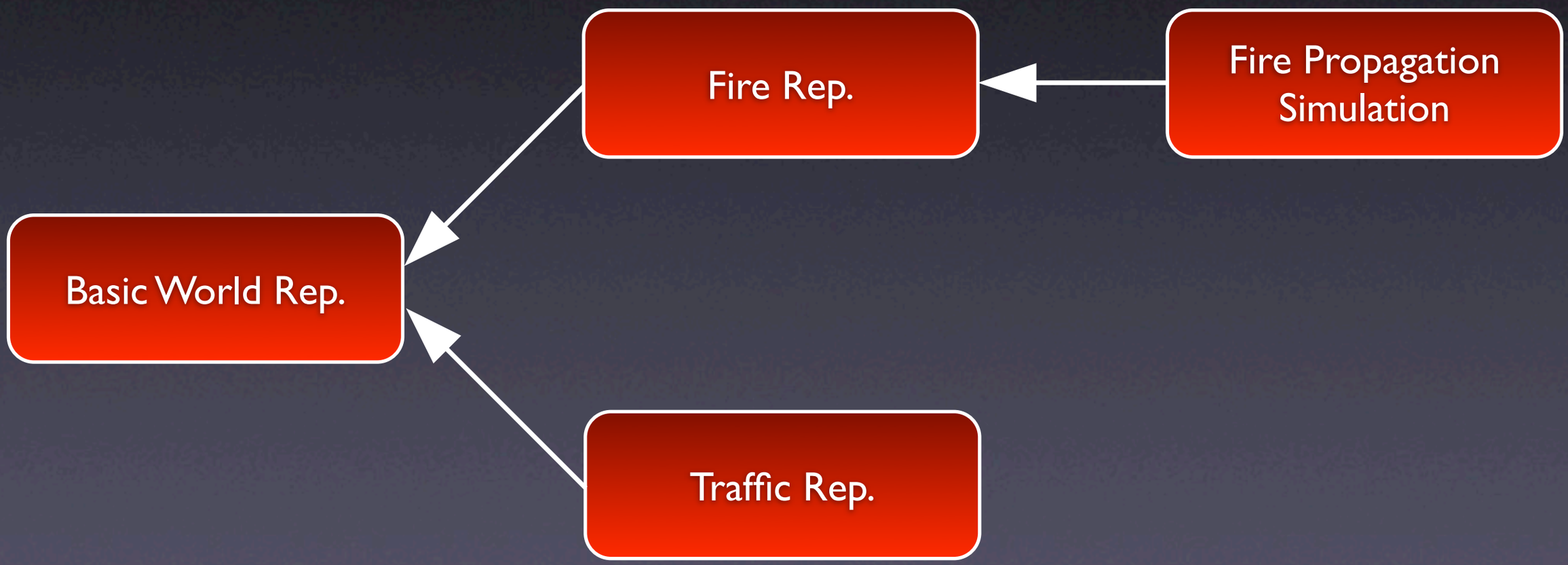
- Existing micro-agents:
  - comm-r, comm-s: RSS message reception and sending;
  - main: handles the agent startup and maintains some agent-related info;
  - basic-world-model: basic world representation;
  - decision: decision micro-agent template.

World representation

- It's necessary to manage several different world representations.
- Each representation must be coherent with others.

World representation

- Solution: representations graph



Tools

- Graphic world representation;
- Loggers.

Tools

- Graphic world representation
- Useful to check the agent internal state.
- Much better than browsing trough thousands of objects…
- Built-in representation: Latex.

World Representation

Start-time: 0 **Elapsed cycles**: 6
Longitude: 0 **Latitude**: 0
Wind force: 0 **Wind Direction**: 0

Loggers

- Easy creation of log messages.
- Log output may be a file or the command line.

Documentation

- The framework has extensive documentation:
  - Conceptual – framework architecture and inner workings, ideias behing its conception, etc…
  - Reference – Classes, APIs, etc…

Why does LISP kicks Java ass?

LISP vs. Java

- The Java language lacks expressiveness and power on many aspects, because it was made based on the idea of attracting the C++ programmers.

LISP vs. Java

- LISP is a programmable programming language:
  - You may create LISP Macros (in LISP!) that generate LISP code.
  - Full expressive power on the Macro construction.
- Java Macros? :)

LISP vs. Java

- CLOS:
  - Multiple inheritance;
  - Multiple dispatch;
  - *before:*, *around:* and *after:* methods;

LISP vs. Java

- LISP allows the programmer to change and fix the code while the program is running, to hold the program execution, debug, backtrack, etc…
- Avoids many fix-compile-run cycles.

Topics

- Introduction
- Domain
- Existing frameworks analysis
- The CLRF framework
- Building an agent
- Conclusions

Building an agent

1. Build additional world representations
2. Build additional micro-agents
3. Build decision micro-agent
4. Update the dispatch table

Tópicos

- Introduction
- Domain
- Existing frameworks analysis
- The CLRF framework
- Building an agent
- Conclusions

Feature	*CLRF*	*yabapi*	*RescueCore*
Thread based structure	Yes	No	No
World representation	Modular and extendable	Basic	Basic
Graphic world representation	Yes (LaTeX)	No	Yes (X-Windows, support for layers)
Log generation	Yes	No	Yes
Documentation	Conceptual and reference	No independent documentation	Just a very incomplete JavaDoc
Programming language	LISP	Java	Java

Future work

- Basic world representation
- Message reception and sending
- Path planning
- Fire representation

Q&A

Miguel Arroz
arroz@guiamac.com