Control Architectures of Robots 2008

A simple architecture for modular robots

Aurélien Godin
DGA/ETAS
aurelien.godin@dga.defense.gouv.fr

Contents

Origins of the project

Triggering event…

- Candidates on our own to the first European Land Robotics trial – Elrob'06
- A simple software was needed to
  - Communicate with the control station
  - Gather a few functional modules (teleoperation, video, localisation, battery management)
- Robustness of the solution had to be improved but was quite operational

... and other issues

- Robot crash
  - Original commercial software did not work anymore on the upgraded system
- Difficult capitalisation
  - No means to gather all students' works
  - Most students with no specific robotics knowledge have difficulties to apprehend usual tools and associated concepts (Player/Gazebo, MissionLab, Carmen, etc.)
- Demonstrations to officials
  - The system to be developed for the robot should be reliable
  - Should be representative enough of state of the art trends so that demonstrations to our visitors are convincing

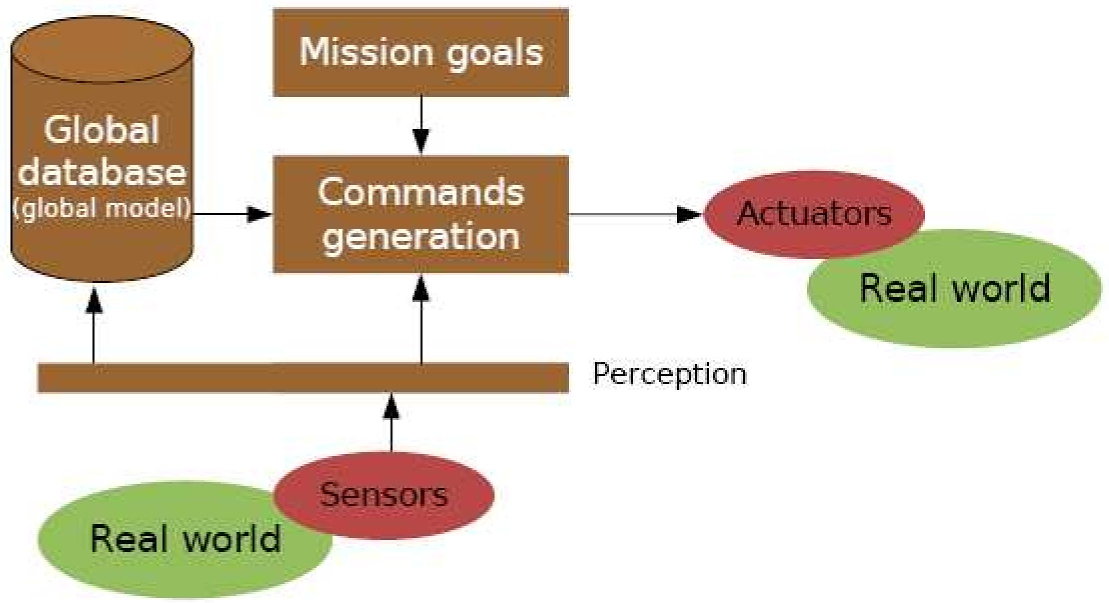
Specification

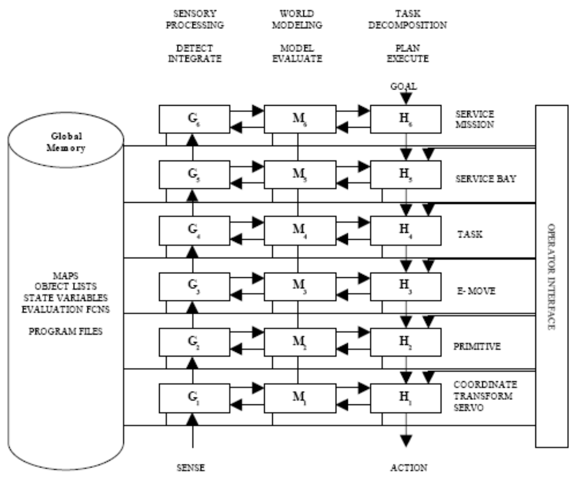
- Modular design
  - To capitalise students' works
  - To get the robot more efficient while technologies evolve
- Robustness
  - One process crashing must not bring the whole system down
- Ease of use
- Representativity
  - Proposing totally new concepts is not our goal…
  - … reusing results available in the literature might not be a so bad idea

⇨The existing Elrob software, complemented with well-functioning concepts described in the bibliography, was an interesting base

Trends in architectural design

Deliberative approaches



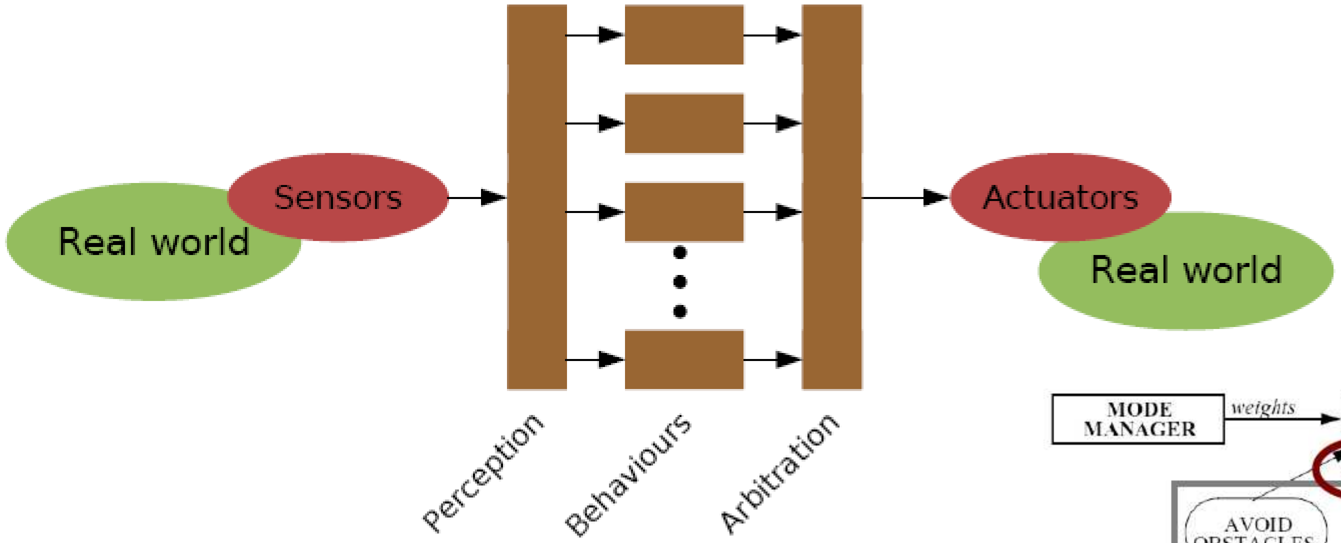


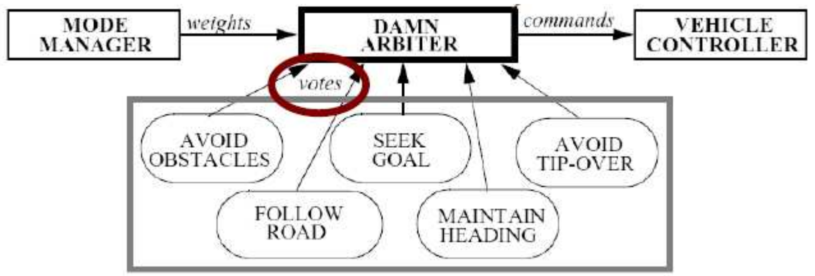
NASREM example [Albus87]

- “Goal-driven” architecture
- Takes benefit from a global world model

- A high hierachisation leading to poor reactivity

Reactive-behavioural designs





DAMN example [Rosenblatt97]

- Tasks parallelism
- Short commands cycle

- Straighforward (local) relation between perception and actuation… leading to dead-ends

Hybrid architectures advantages

- Bring all the required flexibility
- Enable to build robots that both include
  - Reactive capacities (in mobile robotics, to ensure local safety)
  - Deliberative processes, to optimise the actions in function of a high level goal
- Experiments and recent challenges (e.g. DARPA Grand Challenge) have shown very promising results using this kind of designs

Military missions preparation

- In the first moments, no specific doctrine will be created for the use of robotics devices
- Robots should then integrate quite transparently the current combat organisation
- All units, today, act based on a plan that is declined, through geographical information systems, to each level

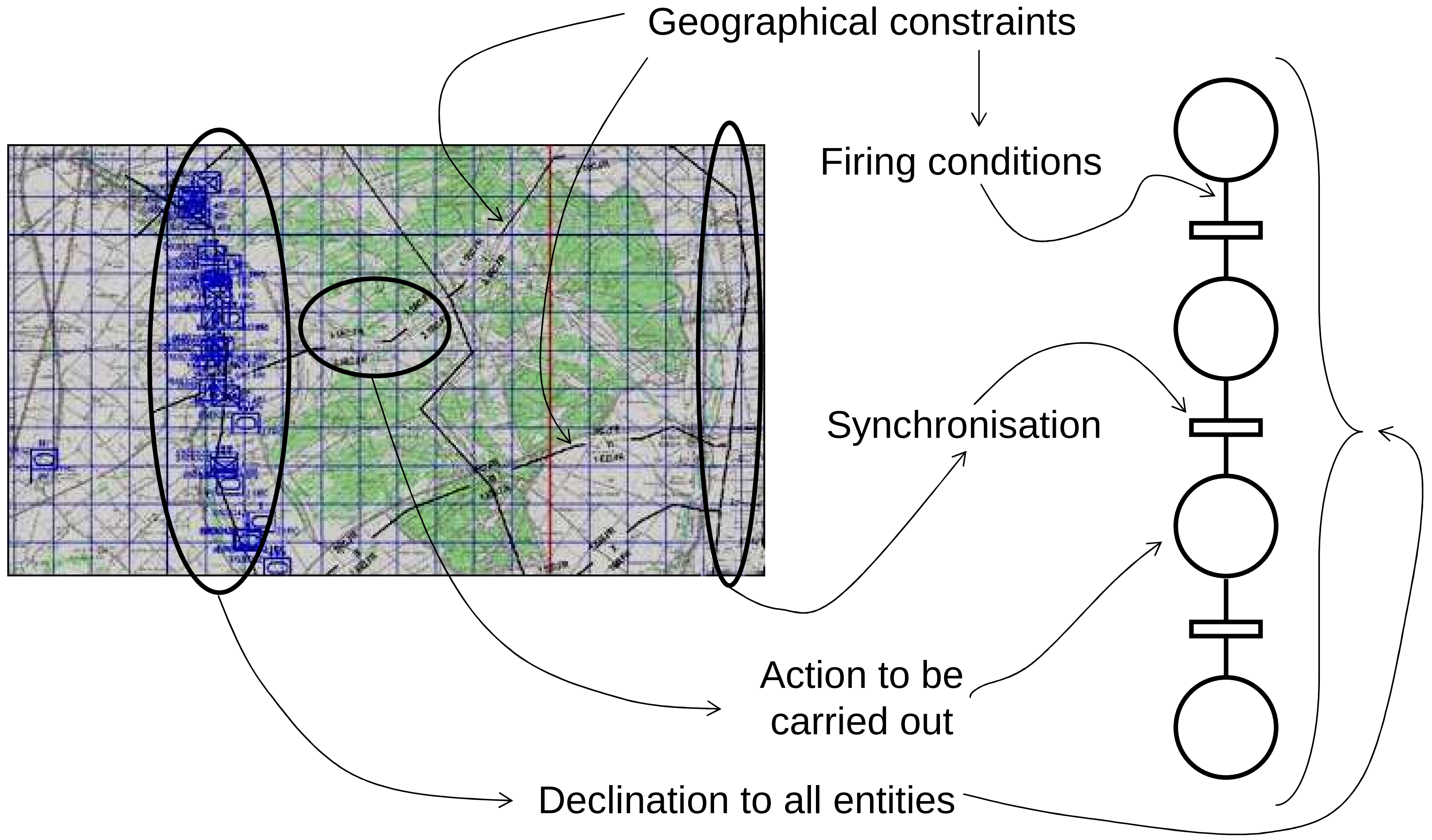
⇨Robots should be able to receive, handle, understand military-like plans, describing

. Geographical constraints

. Time conditions and limits

. Tactical situation information

Analogy with Petri Nets formalism



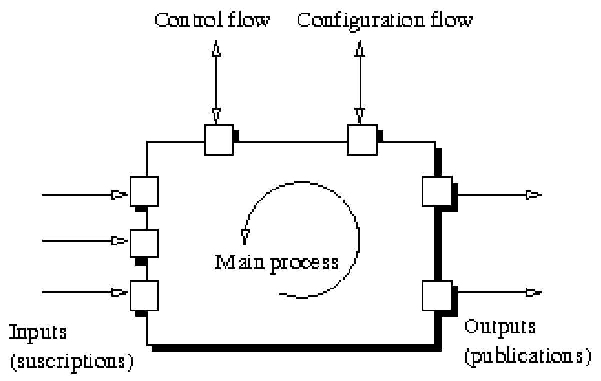
ArMoR in review

Choices retained

- Two-layered structure :
  - Functional modules + execution control level
- Independent processes
- Direct (socket-based) links between them
  - Distribution is possible
- Automatic detection of links rupture and re-connection
- Simple Petri Net-based controller

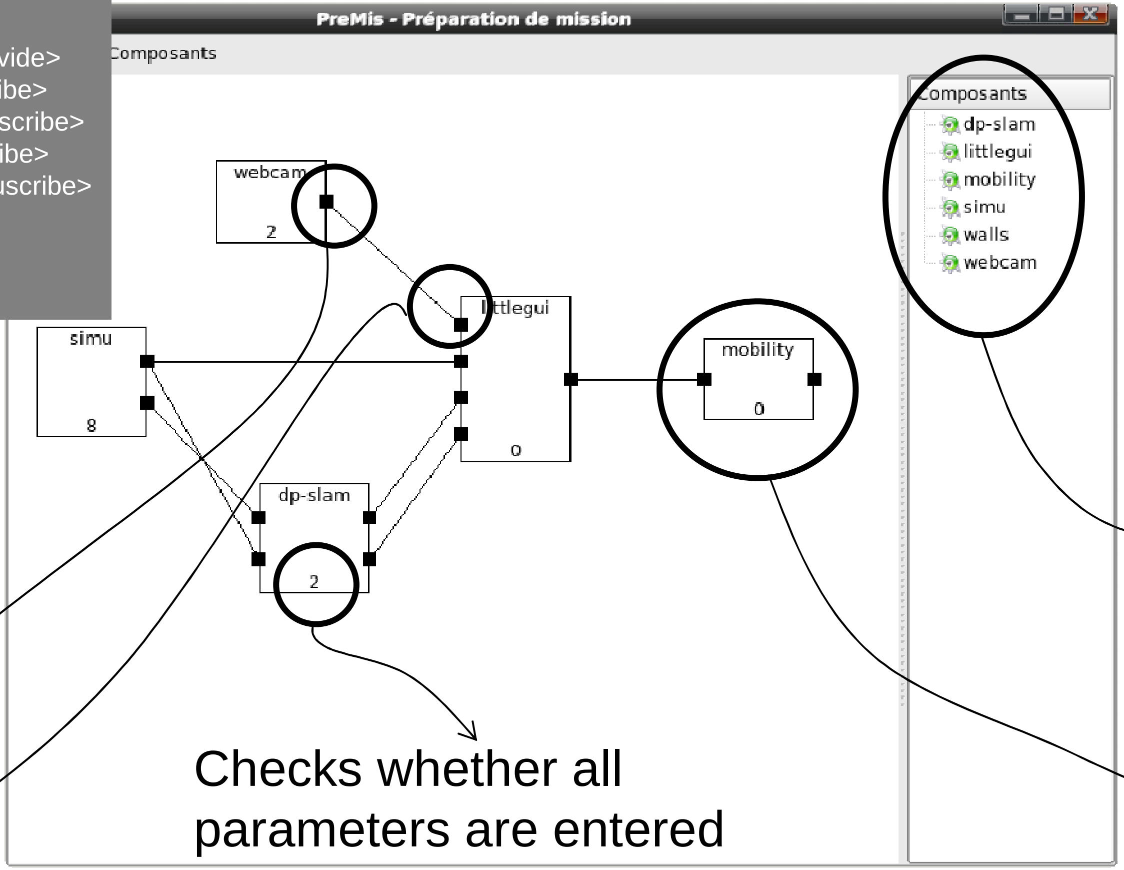
Modules model

- Component-based approach
- Data exchanges
  - Unidirectional ports
  - “Push” flow model
- Separate control channel
  - Execution control
  - Reports from the module
- Configuration port
  - A global configuration server
  - Automatic update of the server while running (Linux only thanks to ‘inotify’ utility)



Organising the modules

```
<armor>
    <component>
        <littlegui id="0" >
<provide id="mobility" >NAV_0</provide>
<suscribe id="video" >VID_0</suscribe>
<suscribe id="laser" >LASER_0</suscribe>
<suscribe id="map" >MAP_0</suscribe>
<suscribe id="pixelLoc" >LOC_0</suscribe>
        </littlegui>
        <mobility id="0" >
…
```



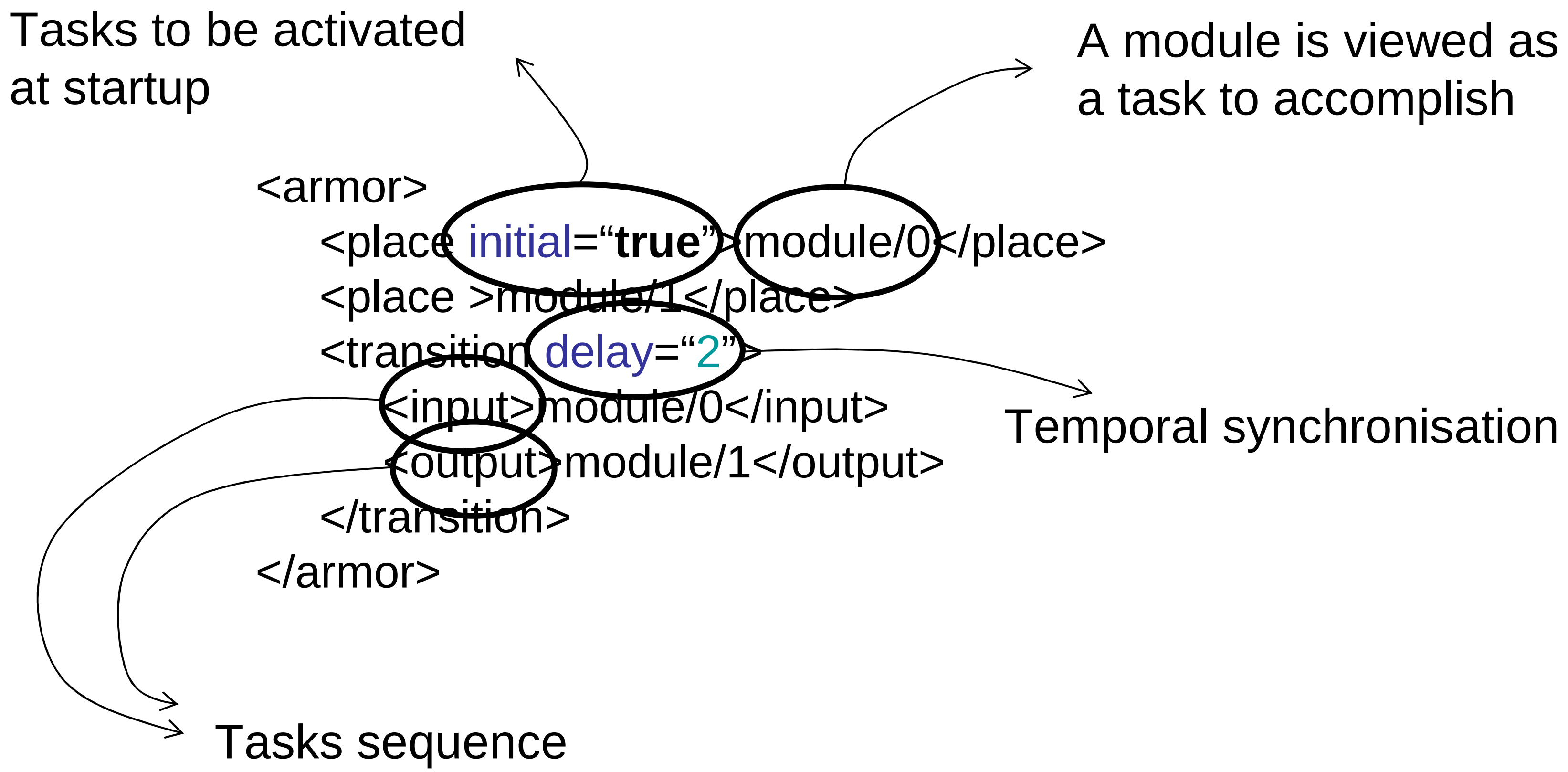
Available modules

Checks whether all parameters are entered before writing file

Data compatibility check is performed during diagram construction

Run in configuration mode, modules provide inputs and outputs number, parameters number

Entering the mission plan


Tasks to be activated at startup
A module is viewed as a task to accomplish
Temporal synchronisation
Tasks sequence

```
<armor>
    <place initial="true">module/0</place>
    <place >module/1</place>
    <transition delay="2">
        <input>module/0</input>
        <output>module/1</output>
    </transition>
</armor>
```

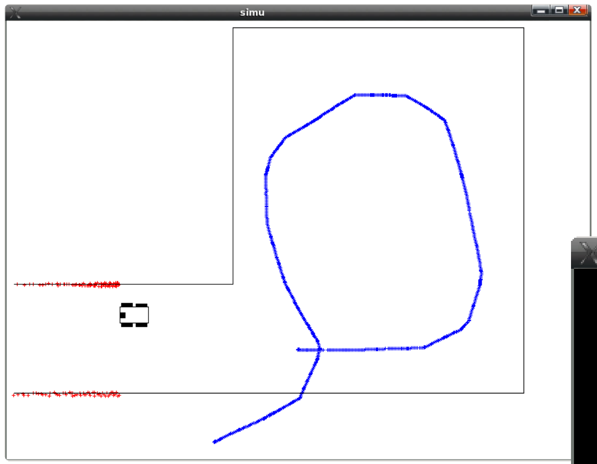
Already operational in ArMoR…
… but still very basic

First experiments

First results with SLAM

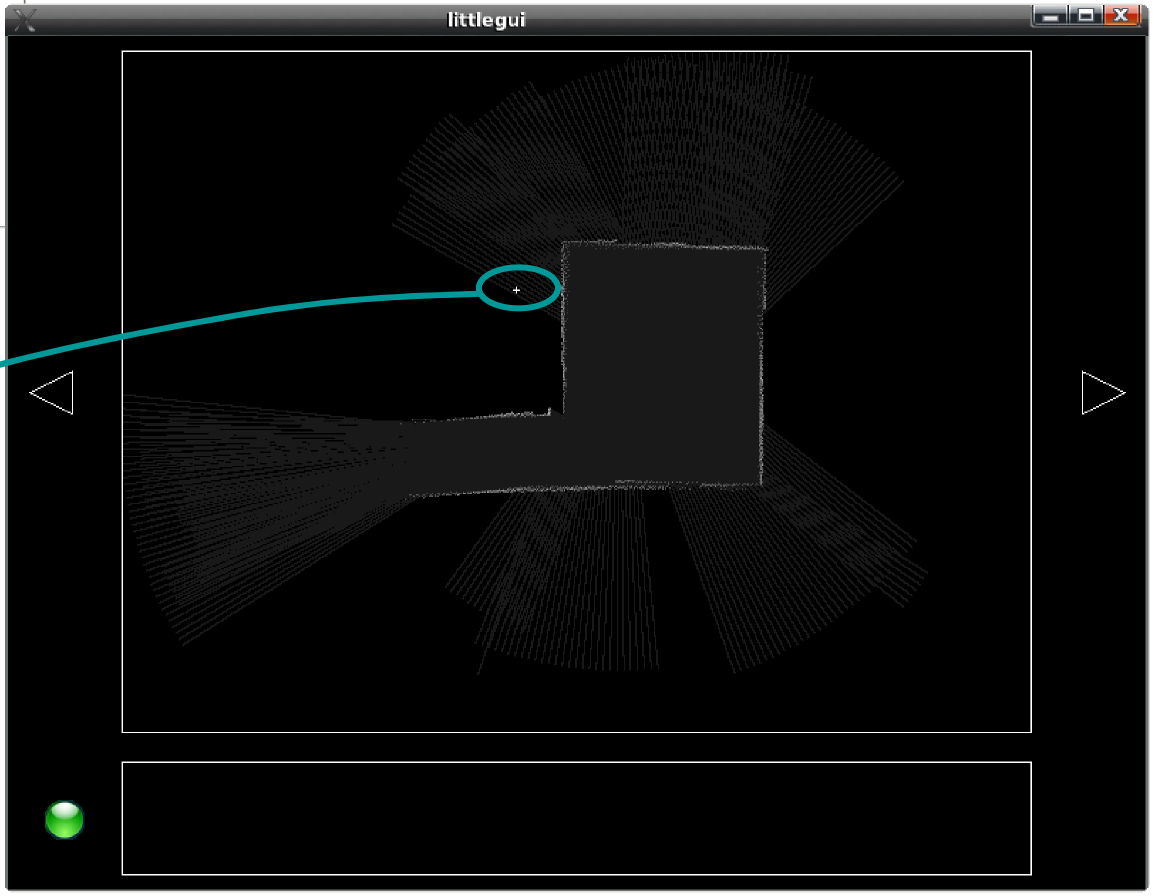
- Ideas : find a module
  - that we did not really understand … so that we were unable to tune it to the architecture
  - that generated large pieces of data
  - that was resources consuming
  - (if we managed to get it work,) that is representative of robotics efforts and can be shown within demonstrations
- SLAM was the ideal candidate
  - DP-Slam [Eliazar, Parr, 2004]

Snapshot of the experiment



The simulator view…

… and the encouraging mapping process result



Even if some unexplained phenomena still persist (maybe is it a quantic robot ?) ;o)

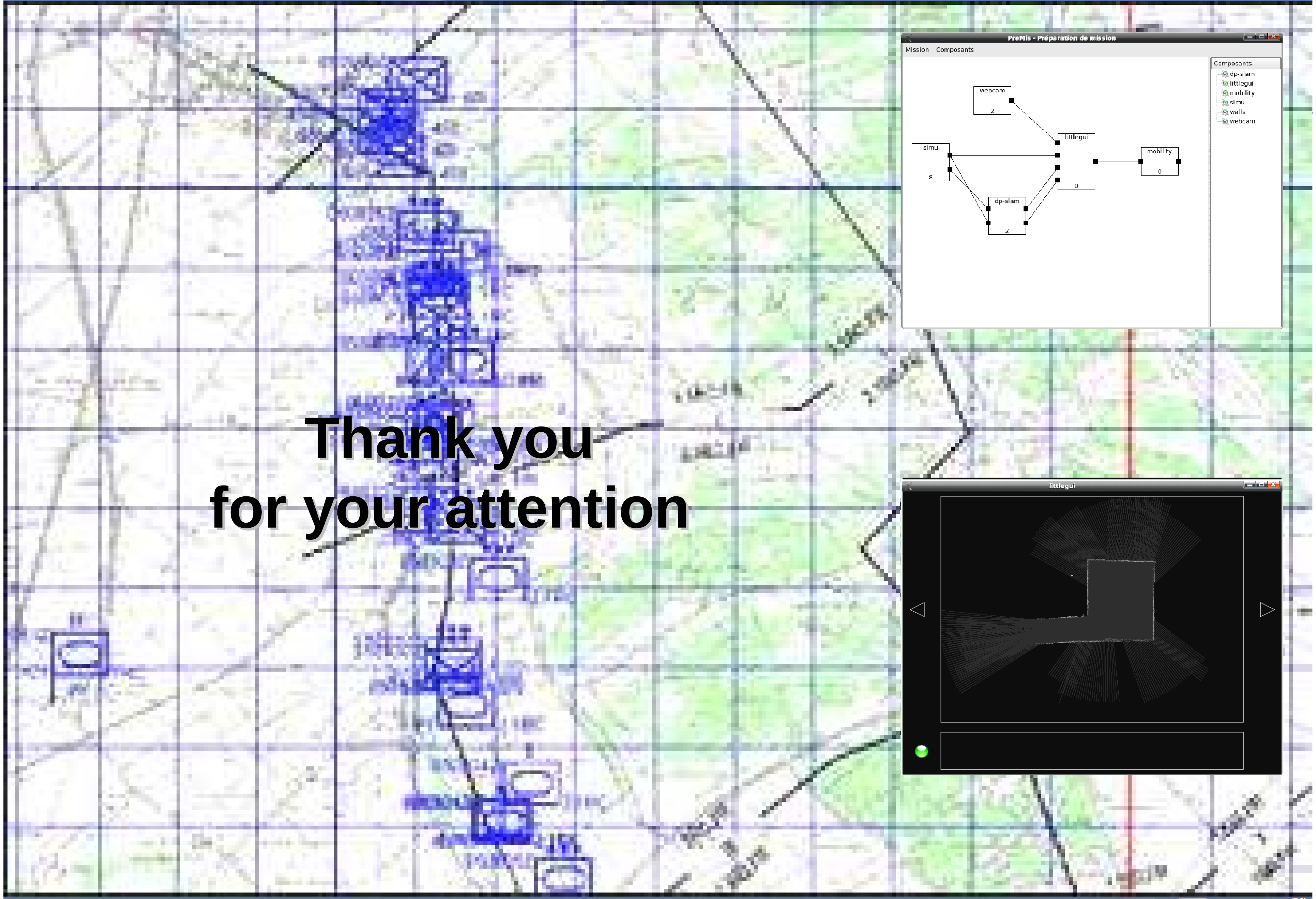
Outlooks and discussion

ArMoR in a nutshell

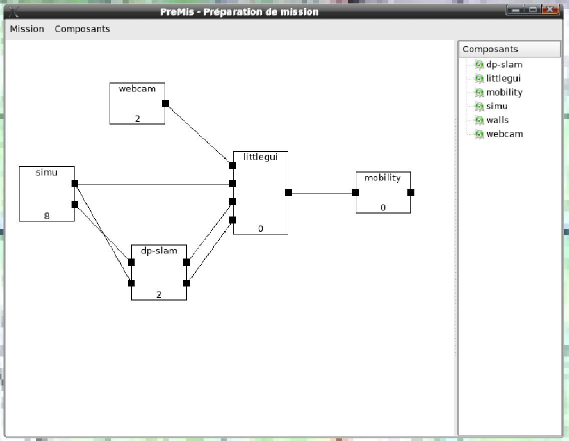
- What it is absolutely not
  - – a scientific breakthrough !
- What it aims to be
  - – a simple framework to capitalise functional modules and offer a common benchmark reference
  - – a demonstration of robotics functionalities
  - – maybe, a simple tool to teach what is an architecture ?

Improvements to include

- Data links
  - adding UDP protocol to extend capabilities (for instance, of teleoperation process)
- Data flows
  - offering the developer the choice between a “push” model and a “pull” one
- Robustness
  - adding a watchdog mechanism
- Control
  - extending controller functionalities
- Graphical tools
  - developing the cartographic $\leftrightarrow$ petri-net-description conversion tool



Thank you for your attention


PreMis - Préparation de mission
Mission
Composants
webcam
2
simu
8
littlegui
0
mobility
0
dp-slam
2
Composants
dp-slam
littlegui
mobility
simu
walls
webcam