ONERA

ProCoSA : a software package for autonomous system supervision

Magali BARBIER, Jean-François GABARD / ONERA Toulouse Centre
Dominique VIZCAINO, Olivier BONNET-TORRES / SUPAERO

Outline

[1] Autonomy

[2] ProCoSA software package

[3] AUV application

[4] UAV application

[5] UGVs application

Conclusions and future work

Autonomy levels

Embedded function

Operator function

5

decisional autonomy

4

objective definition

3

itinerary definition

2

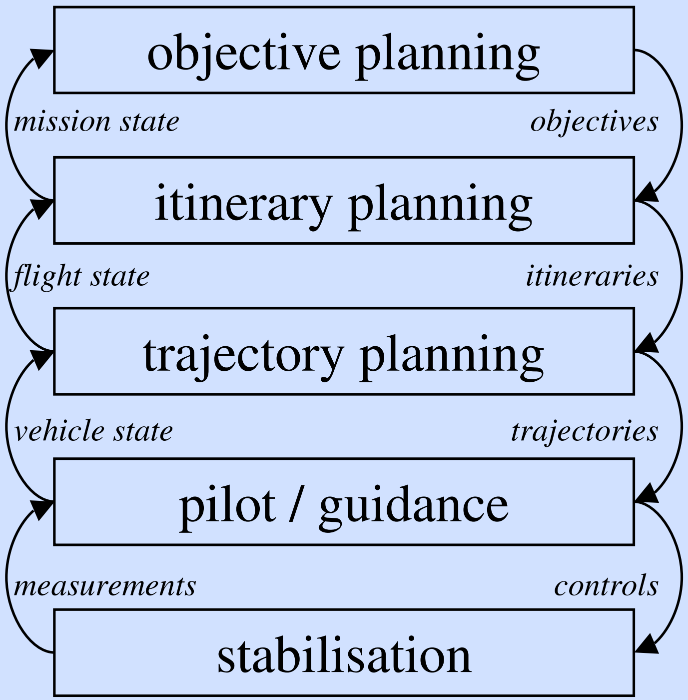
trajectory computation

1

tele-operation

operational autonomy

0



Decisional autonomy

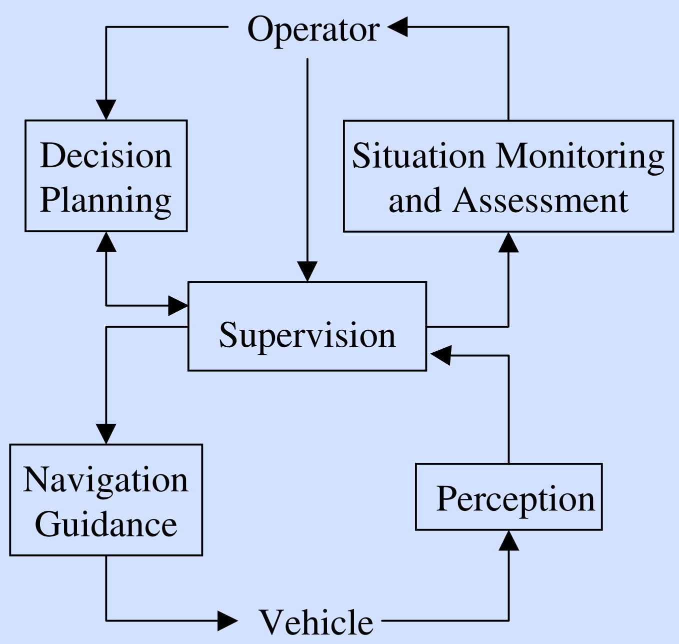
- Partially known, uncertain, dangerous, dynamic environments
- Asynchronous disruptive events, punctual communication failures

- Mission to achieve
  → embedded architecture to online supervise the execution of the mission, adapt mission objectives and initial planning
- Main decisional software program: planning function

- No full autonomy → some tasks and decisions to operators

Embedded decisional software architecture

- Close the loop {perception, situation assessment, decision, and action}
- Supervision function
  - nominal execution (monitoring of vehicle behaviour)
  - reaction to disruptive events
  - run decisional tasks
  - communication with ground station and other vehicles



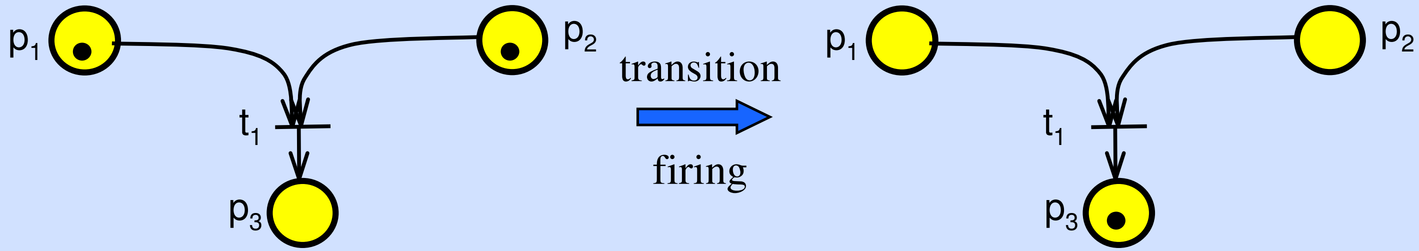
ProCoSA software package

- "Programmation et Contrôle de Systèmes à forte Autonomie" ® 1999
- Integrated package
  - puts together and synchronises functions achieving system autonomy
  - aims at developing an embedded decisional software architecture
- Offline development stages
  - nominal and non-nominal procedures writing: Petri nets model vehicle behaviours
  - software programming
  - co-operation coding between procedures and software programs
- Online execution
  - mission supervision by a Petri net player

[2] ProCoSA

Petri net formalism

- Discrete event modelling
- Graph with { places = state } and { transitions = state modification }
- Marking of Petri nets → state of the system (tokens in places)

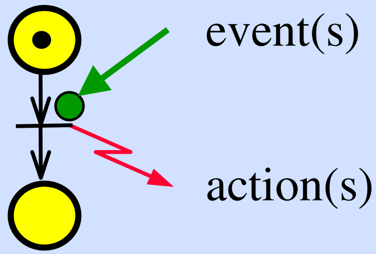


- Sequencing, parallelism and synchronisation representation
- Analysis techniques

[2] ProCoSA

Petri nets in ProCoSA

- Interpreted nets

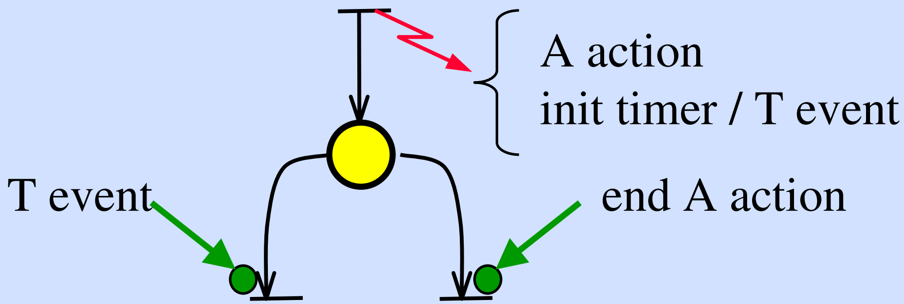


run a net
emit an event
run a software function
...

+ *data flow*

- Firing rules: enabled transition is fired iff one event occurs

- Timers to limit duration of actions



- Hierarchical modelling
  → several levels of detail

The Petri net player

- Tiny language
  - Lisp interpreter for distributed embedded applications (ONERA)
  - Library implementing socket TCP/IP protocol
- Direct interpretation of Petri nets (no code translation)

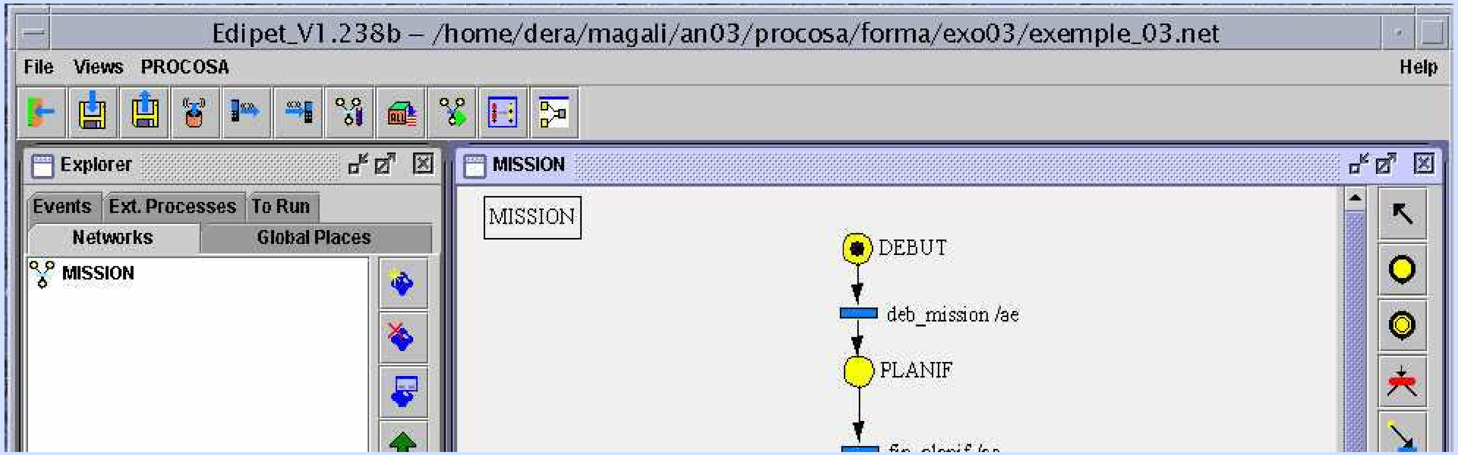
- Supervision
  - procedure execution
    - fires validated transition given the occurrence of events
    - runs actions, e.g. calls the run of a software function
  - dialog synchronisation with software programs
  - communication with external systems (operators, other vehicles)

[2] ProCoSA

ONERA

EdiPet graphical user interface

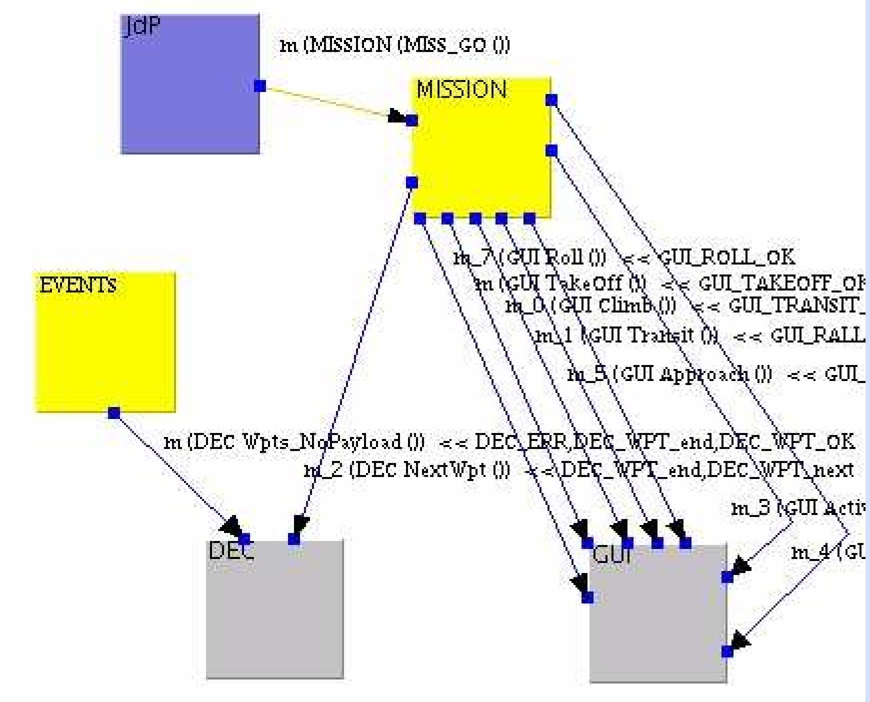
- Project = Petri nets + software function names + relationship
- Offline
  - graphical creation of Petri nets
  - connections inside the project
  - generation of interfaces → skeleton of software programs
- Online
  - display of net states during execution



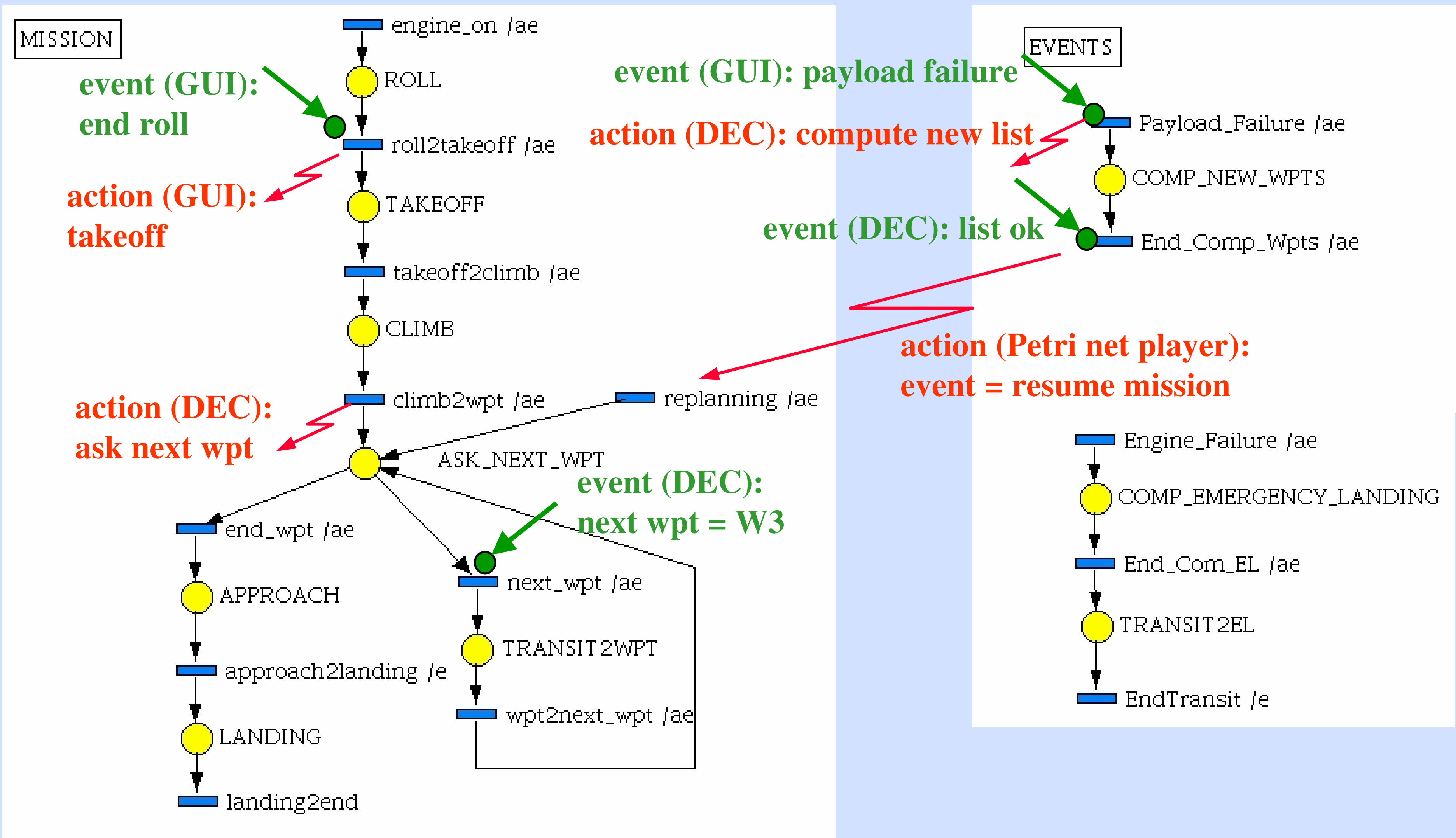
ONERA

Example of project

- Simple mission for an UAV = join a sequence of waypoints
- A waypoint <=> a payload
- Two Petri nets
  - MISSION models nominal phases: roll, takeoff, climb, transits to each waypoint, approach and landing
  - EVENTS models non-nominal reactions to failures
- Two software programs
  - GUI simulates vehicle guidance and payload control
  - DEC
    - next waypoint
    - new list in case of payload failure
    - nearest emergency site in case of engine failure



[2] ProCoSA


MISSION
engine_on /ae
ROLL
event (GUI): end roll
roll2takeoff /ae
action (GUI): takeoff
TAKEOFF
takeoff2climb /ae
CLIMB
climb2wpt /ae
action (DEC): ask next wpt
replanning /ae
ASK_NEXT_WPT
event (DEC): next wpt = W3
end_wpt /ae
APPROACH
approach2landing /e
LANDING
landing2end
next_wpt /ae
TRANSIT2WPT
wpt2next_wpt /ae
EVENTS
event (GUI): payload failure
action (DEC): compute new list
Payload_Failure /ae
COMP_NEW_WPTS
event (DEC): list ok
End_Comp_Wpts /ae
action (Petri net player): event = resume mission
Engine_Failure /ae
COMP_EMERGENCY_LANDING
End_Com_EL /ae
TRANSIT2EL
EndTransit /e

[2] ProCoSA

Verification process

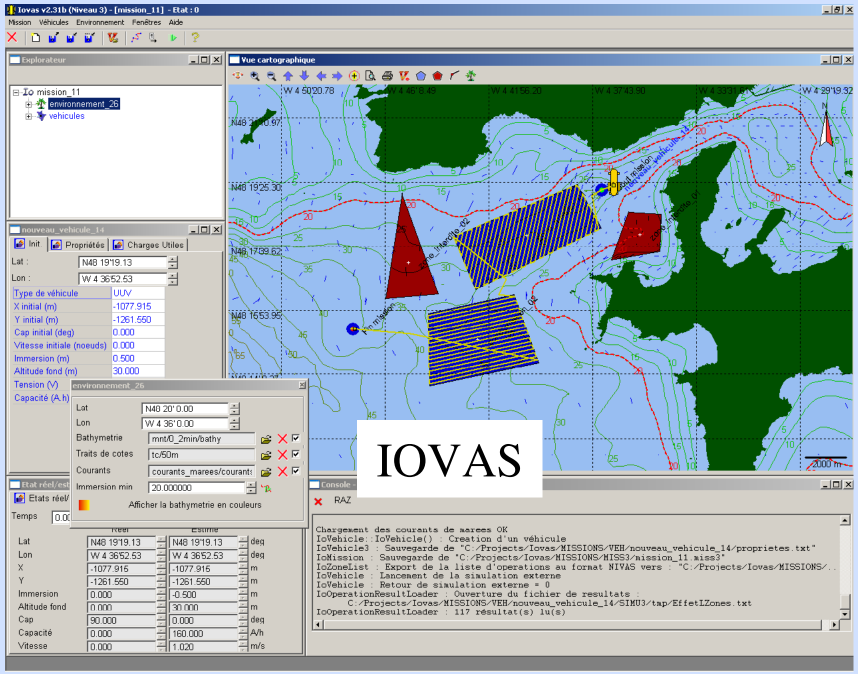
- Available in EdiPet
- Automatic generation of the reachability tree (reachable markings)
  - place safety
  - detection of dead markings
  - detection of cyclic firing sequences

- → Model robustness

ONERA

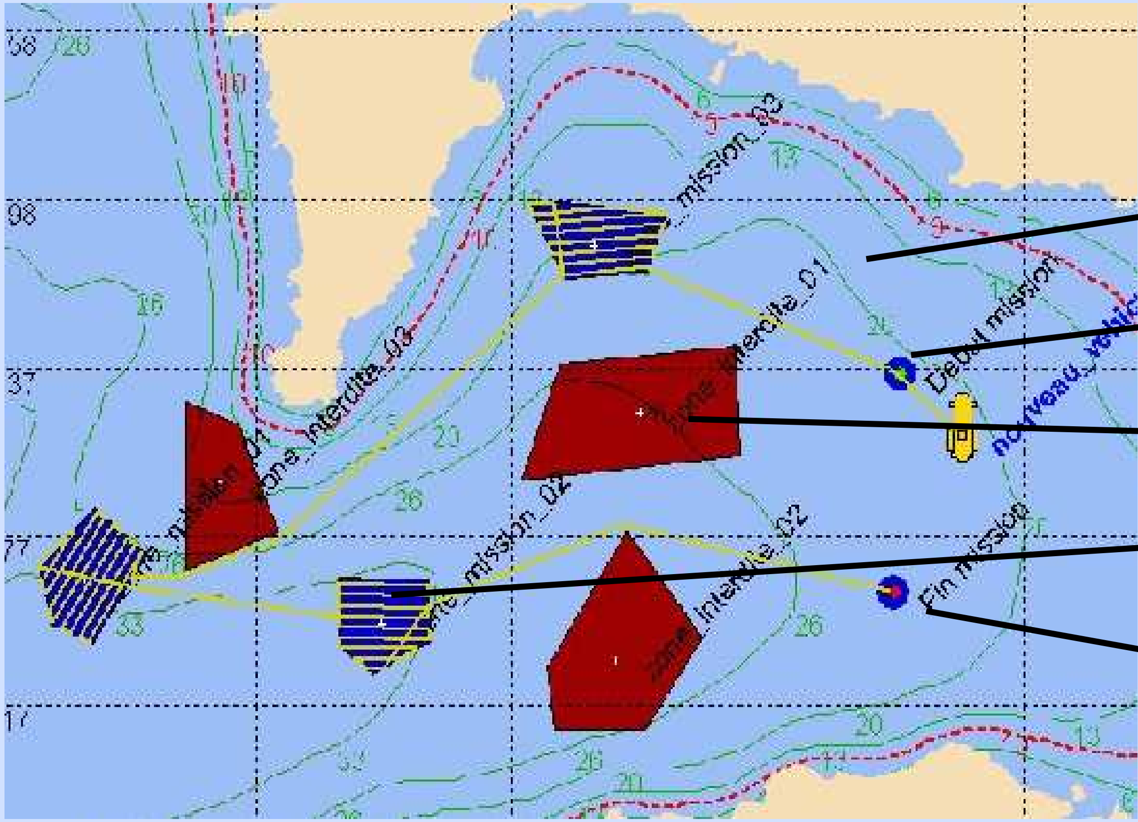
AUV application

- { ONERA + PROLEXIA } for DGA/GESMA
- Demonstration of autonomous missions by AUVs for areas survey
- Redermor AUV, + frontal and side scan sonar
- ONERA: embedded decisional software architecture
  + planning software program NIVAS
- PROLEXIA: man machine interface
  for mission preparation and supervision
- Several autonomy
  levels

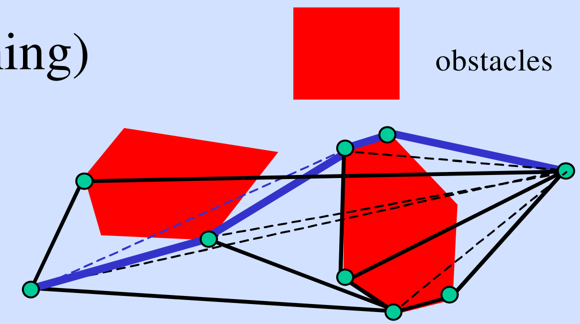


[3] AUV

AUV mission



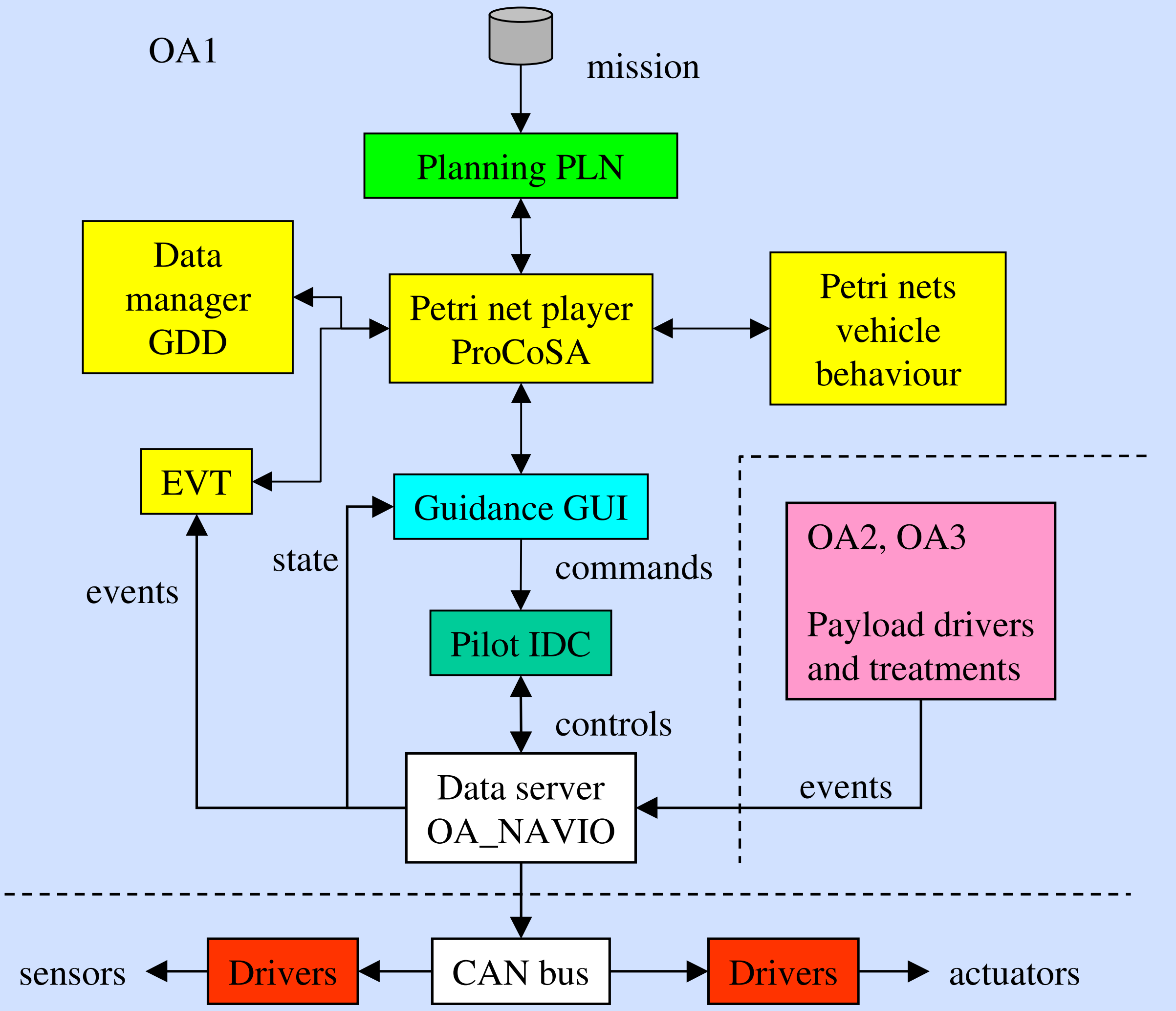
- Autonomy level 3 (trajectory and itinerary planning)
- Planning algorithm
  - visibility graph based algorithm:
  - Little's algorithm → optimised itinerary



Embedded architecture

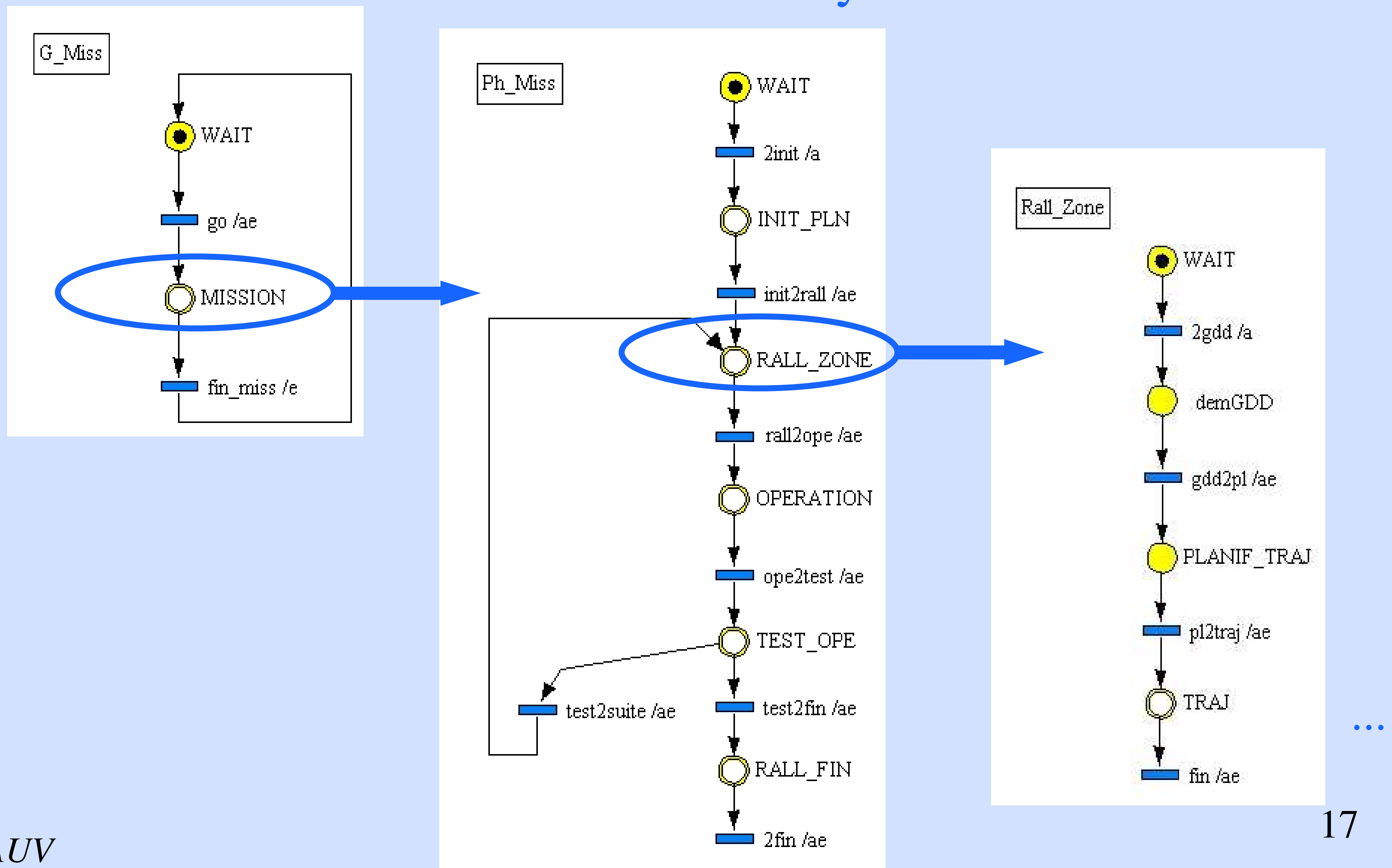
- OA1: decisional computer
- OA2 and OA3: sonar payload control and mine warfare algorithms (classification)

→ Next presentation



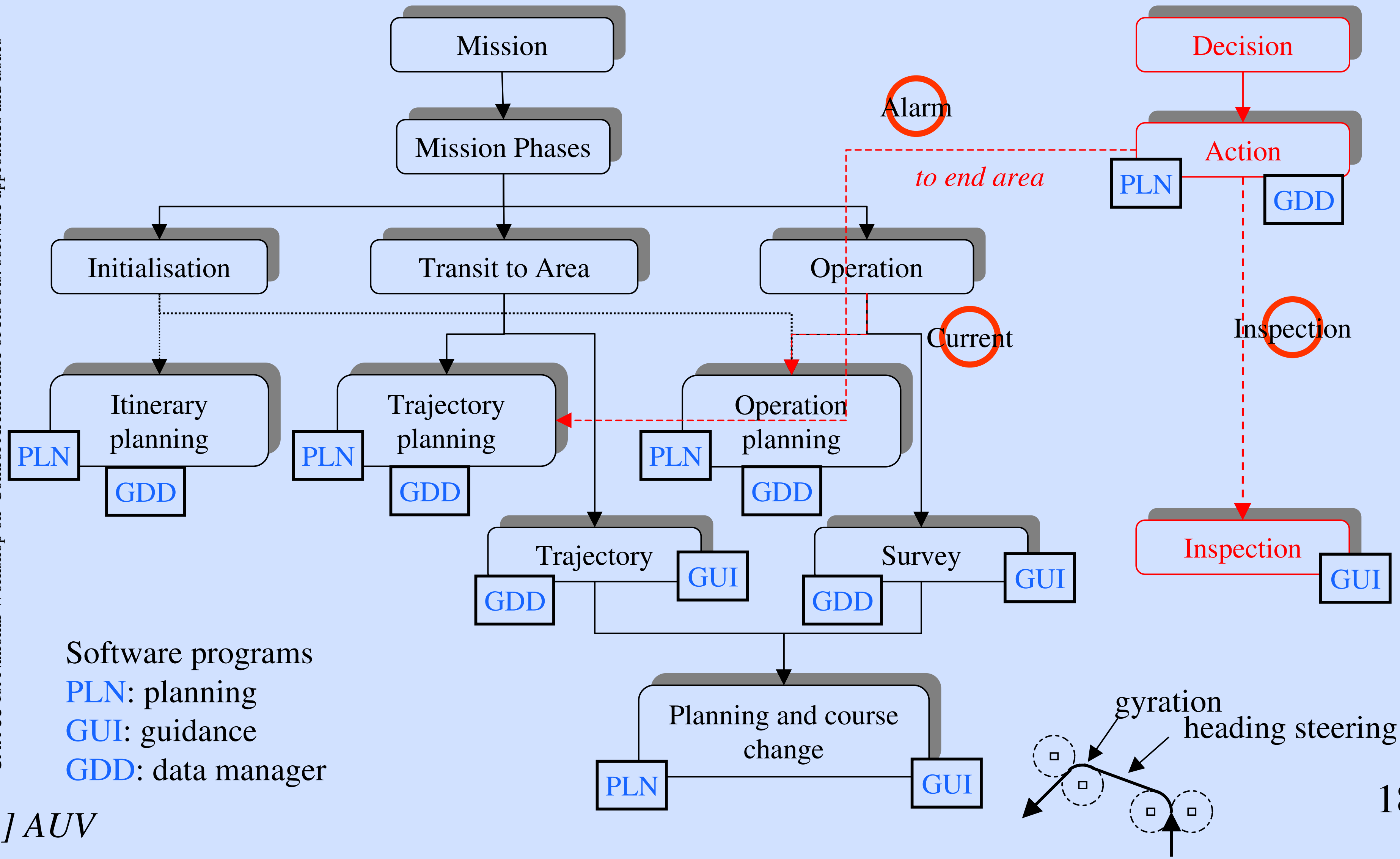
[3] AUV

Petri net hierarchy



[3] AUV

Petri nets for nominal and non-nominal scenarios



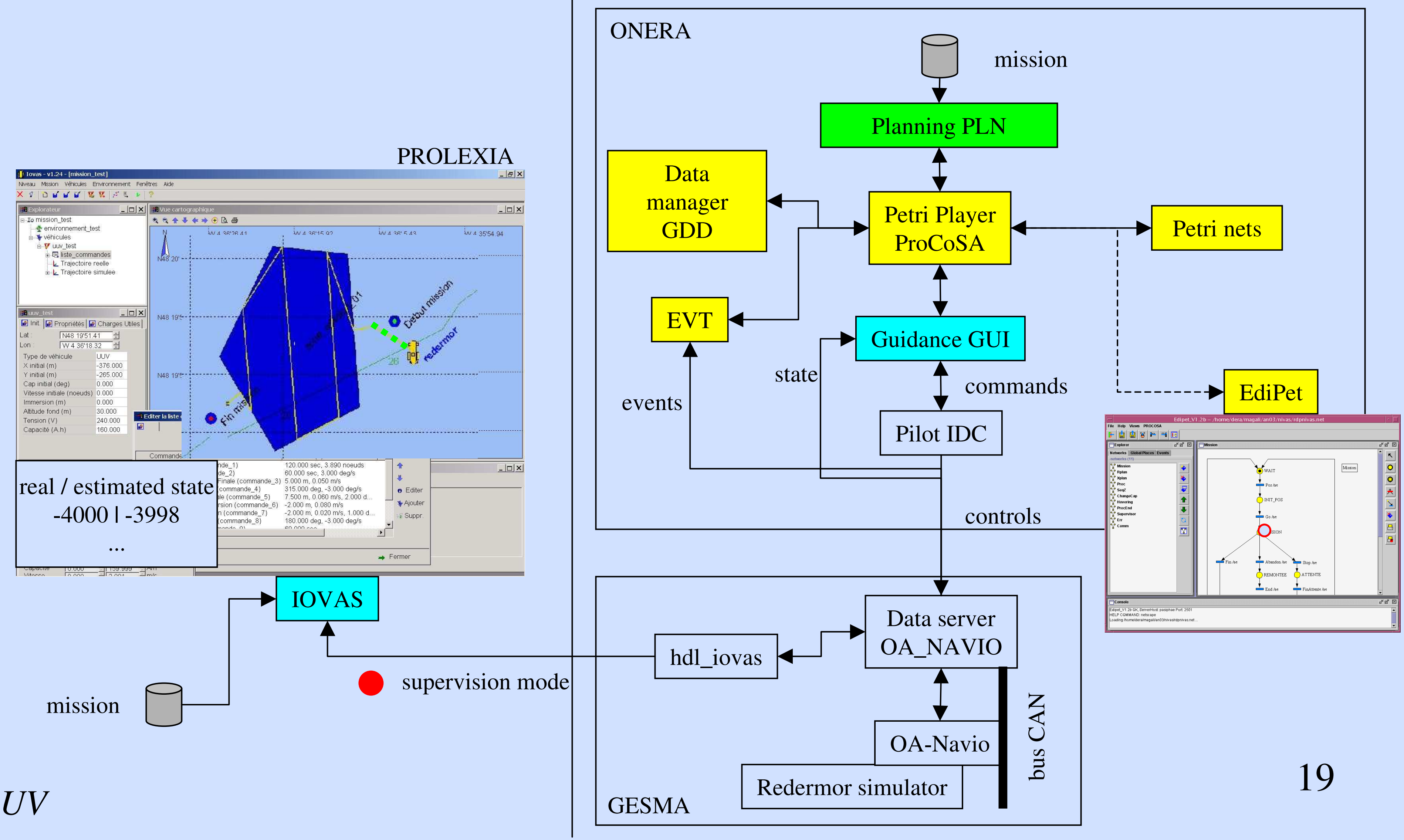
Software programs
PLN: planning
GUI: guidance
GDD: data manager

[3] AUV

Bench test

windowsNT: operator

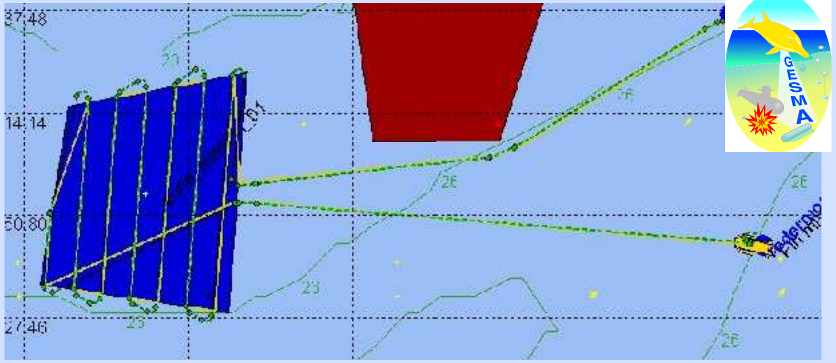
Linux: embedded architecture



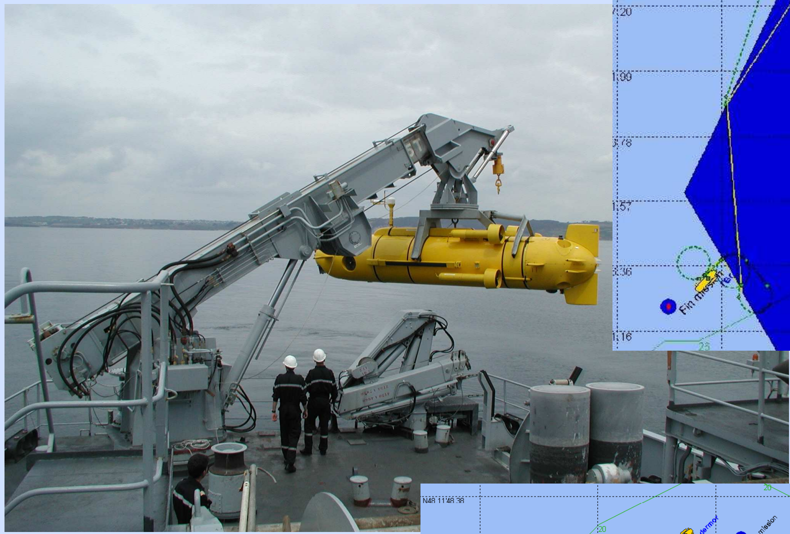
[3] AUV

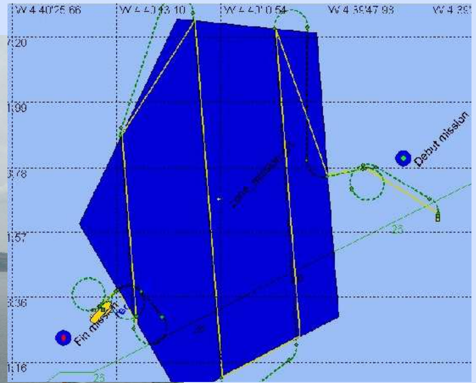
ONERA

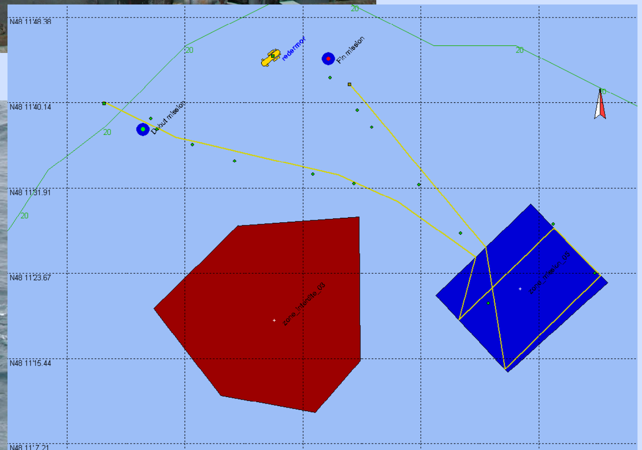
Results



- Lab bench tests
  - several typical missions
  - validation of the whole architecture
- Sea validation of level 3 in Mars 2006: nominal scenarios







[3] AUV

UAV application

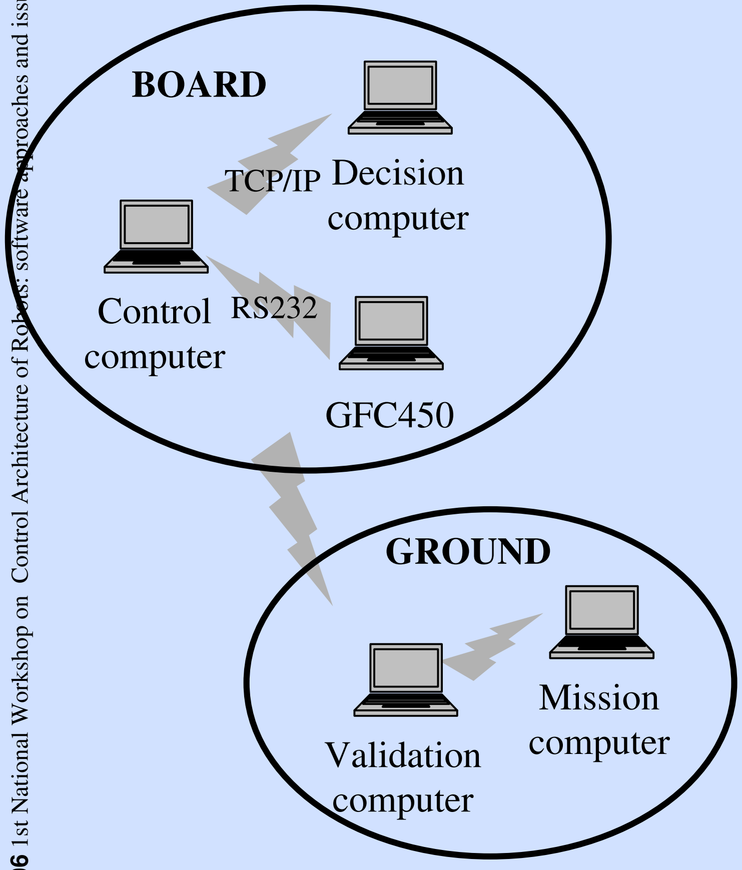
- { EADS + ONERA } project for French Defence Ministry DGA
- Demonstration of an architecture designed for mission supervision
- Generic architecture / { UAV, mission, environment }
- Disruptive internal and external events: failures, weather situation, interfering aircraft, threats… → on-board monitoring and replanning

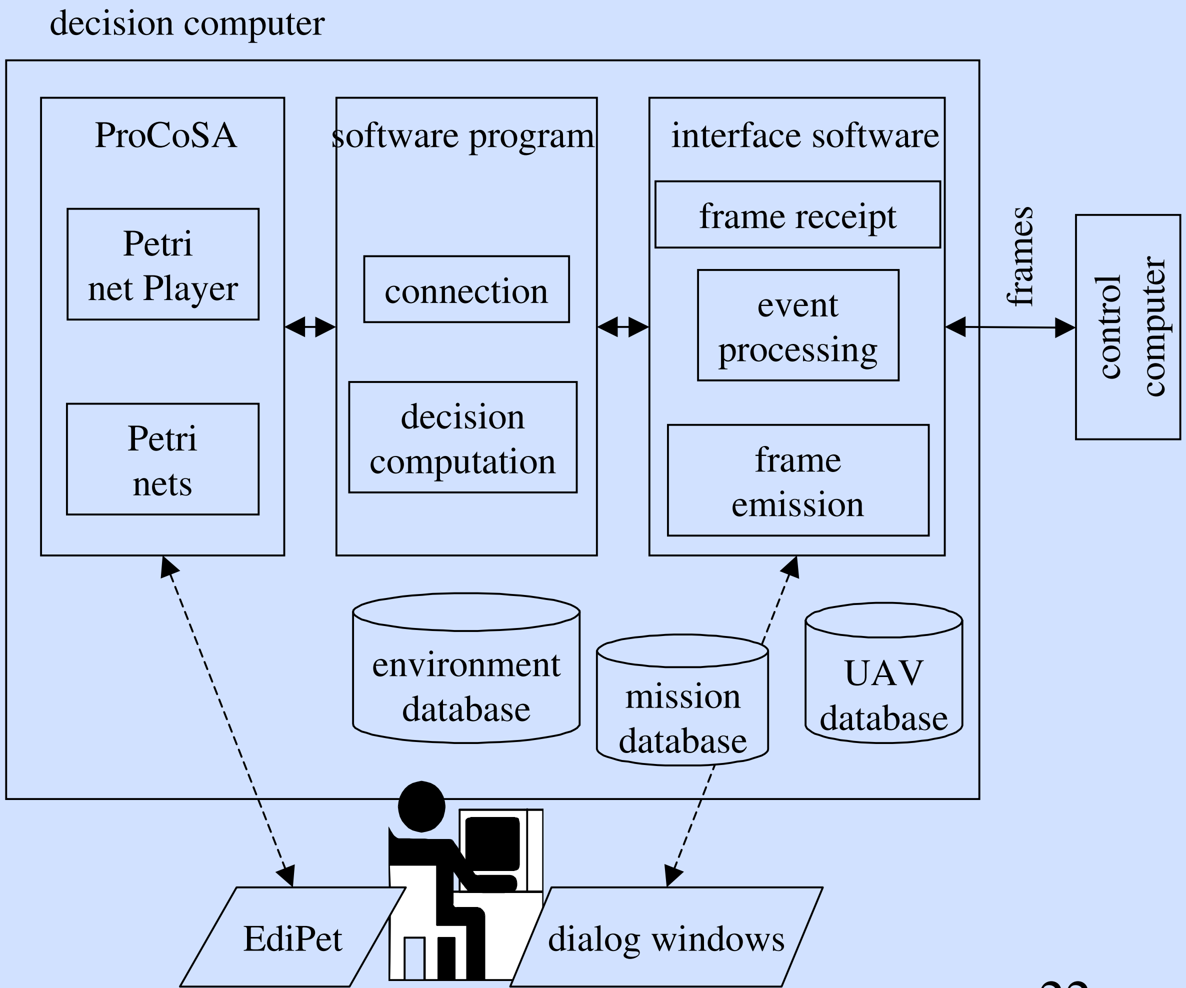
- Mission: join an operation area defined by transit legs and { payload legs = objectives }

- Experiments:
  MCR-4S light aeroplane (DynAero)

[4] UAV

Embedded architecture

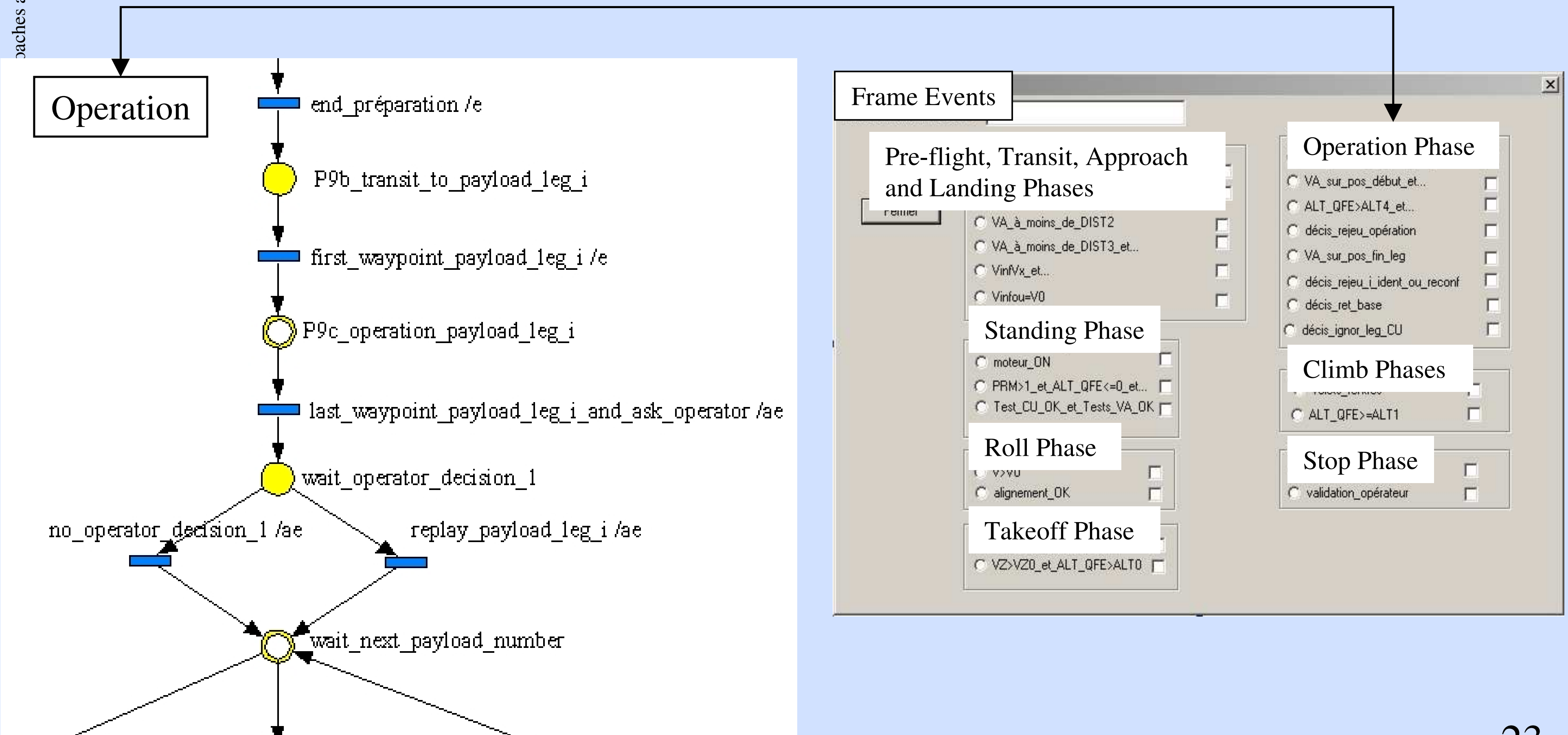




[4] UAV

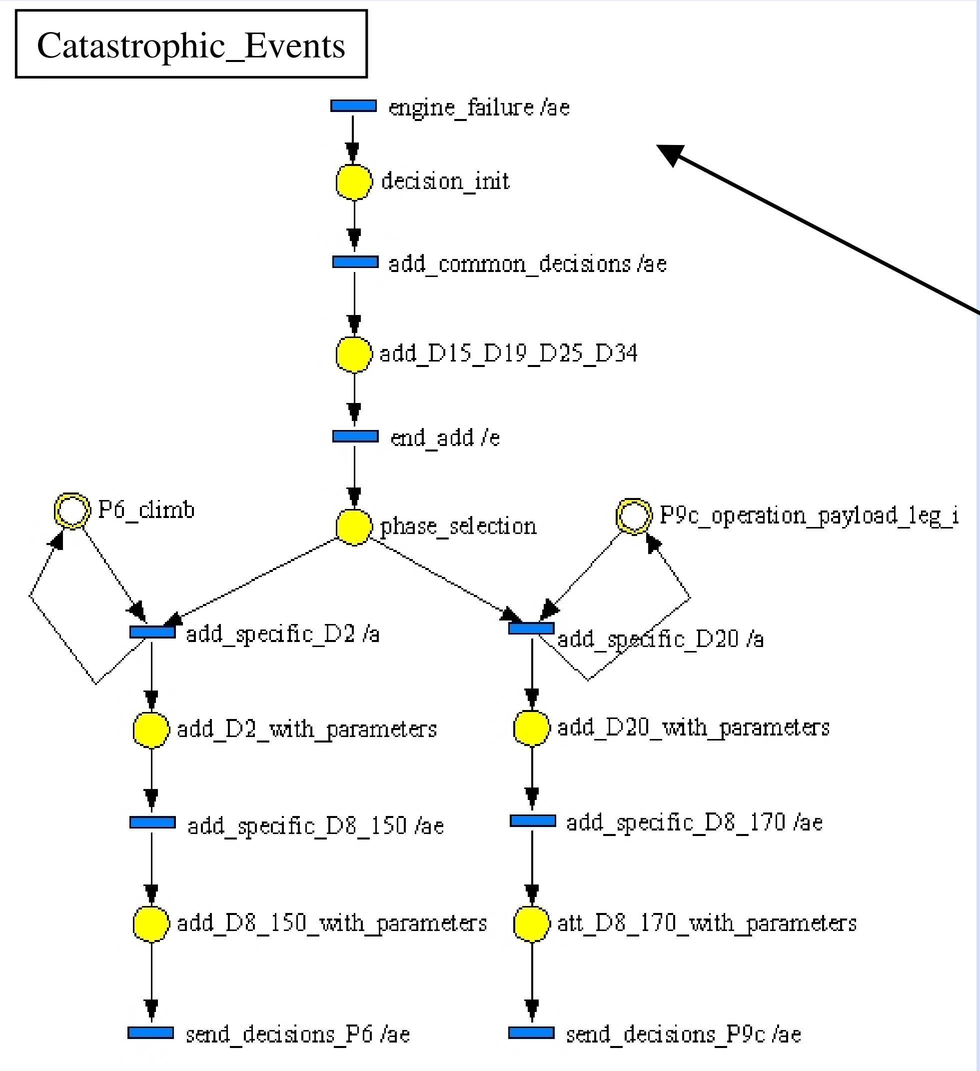
Nominal scenario

- Hierarchical modelling: example of operation level



[4] UAV

Non-nominal scenarios



- Study of disruptive events → classification:
  - catastrophic → mission abortion
  - flight-related → new flight profile
  - mission-related → replanning
  - communication-related → autonomy

- Non-nominal scenarios:
  - list of ordered decisions according to event type and ongoing phase

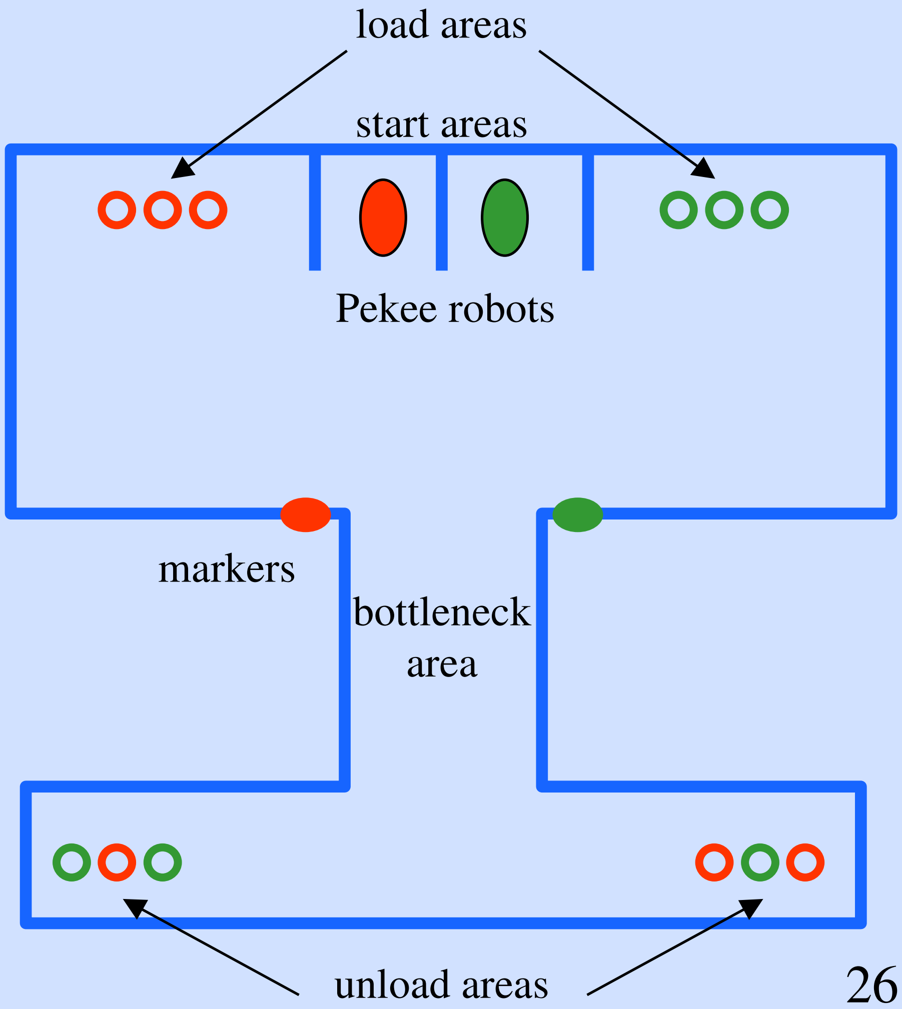
[4] UAV

Results

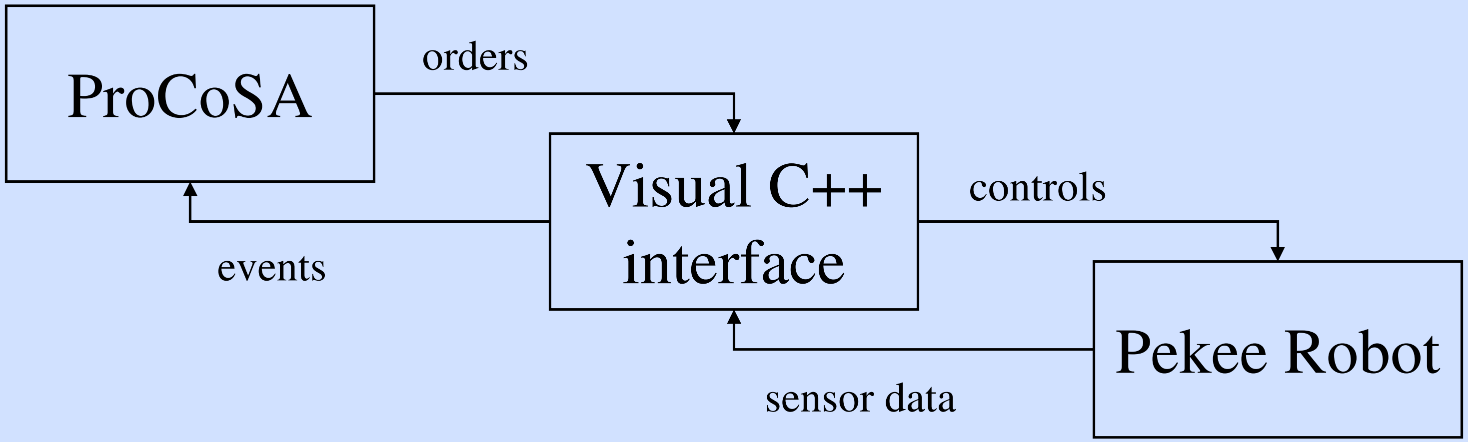
- Formal tests
  - Petri net analysis
- Lab tests
  - simulation tool allows to validate frame transmission in both directions
- Ground and flight tests
  - ongoing: March and May 2006
  - nominal and ten non-nominal scenarios
  - checking of flight data transmission (telemetry, control and operator frames) and Petri nets supervision onboard and on the ground

UGVs application

- SUPAERO students
- Several autonomous robots
- Experiments: Pekee robots
- Mission: known environment, virtual load and unload of rings in a specific order



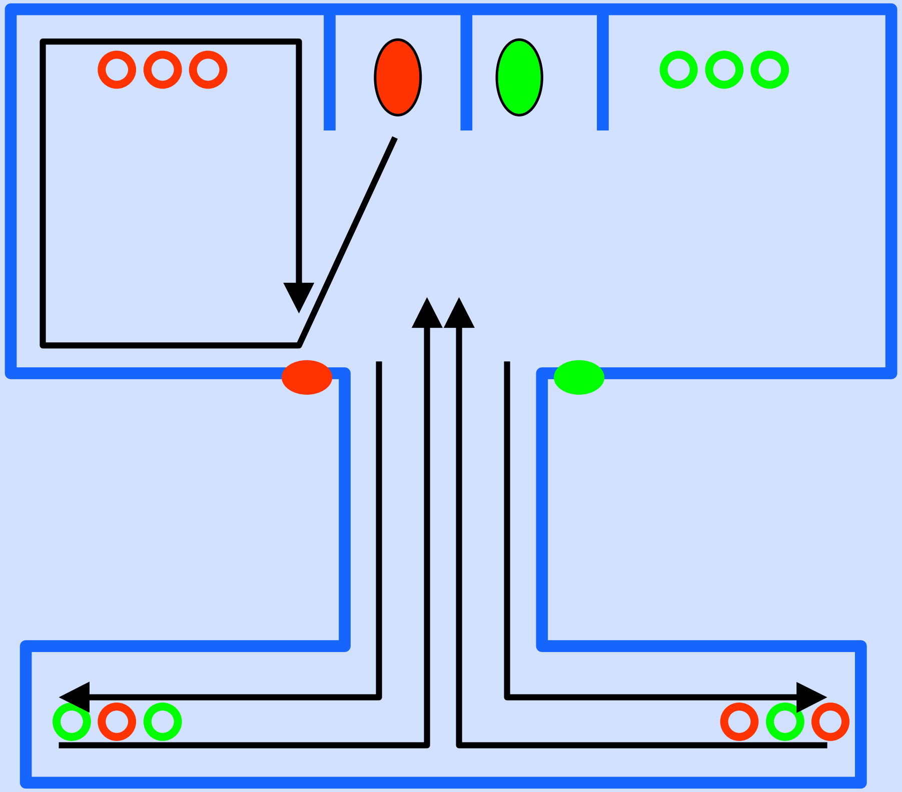
Architecture



- Centralised architecture
- Sensor data: IR and camera
- Elementary orders to move in labyrinth: go forward, go backward, turn right, turn left, follow right wall, follow left wall, enter bottleneck, exit bottleneck

A robot behaviour

- Petri net <=> track
  - dedicated area
  - bottleneck area and load area
    - left
    - right
  - load area and bottleneck area
    - left
    - right
- One Petri net for alternate unloads

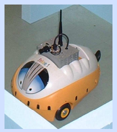


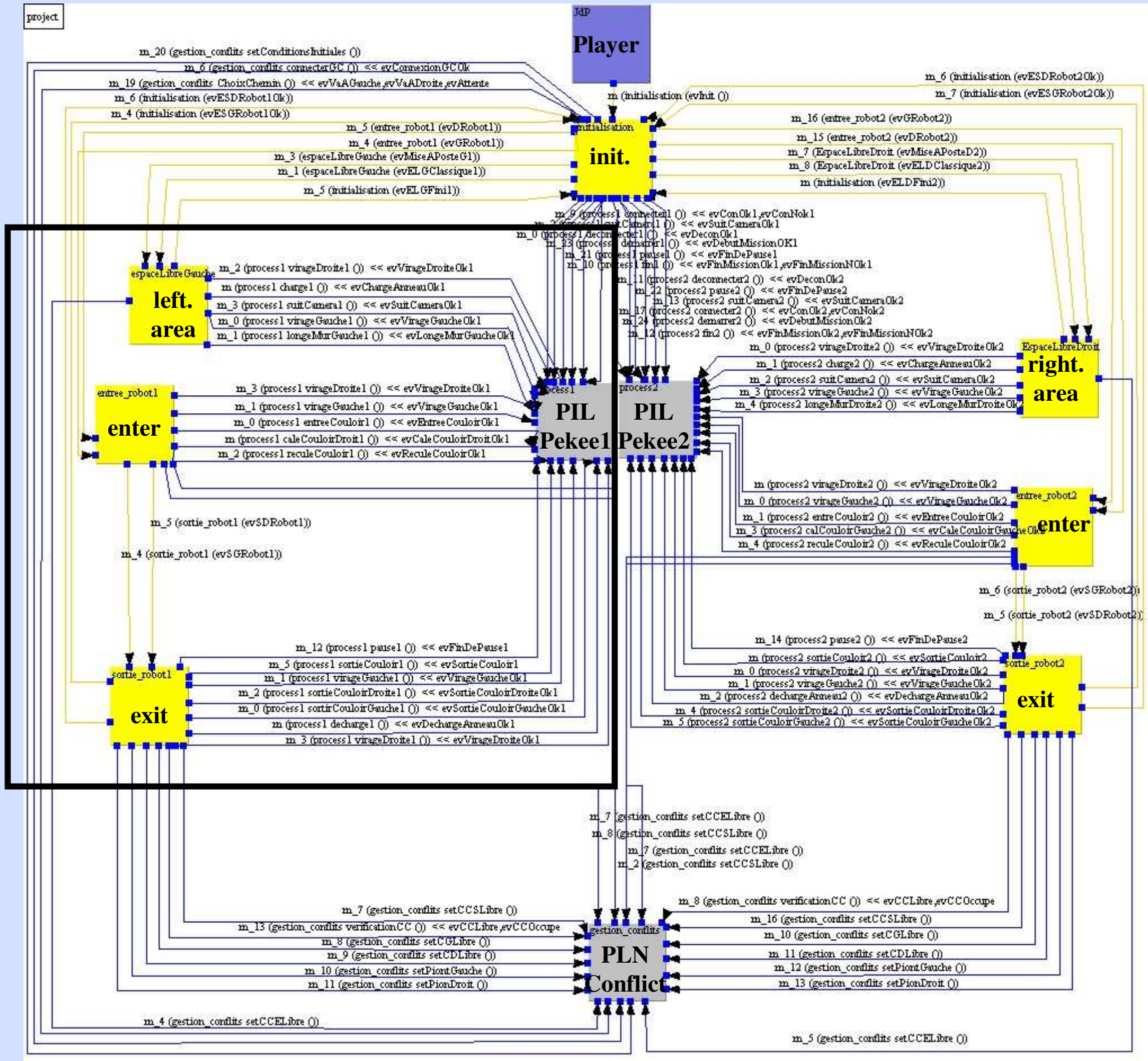
[5] UGVs

Mission execution

- C++ software programs
  - elementary orders
    → Pekee robot: PIL x2
  - planning = conflict management

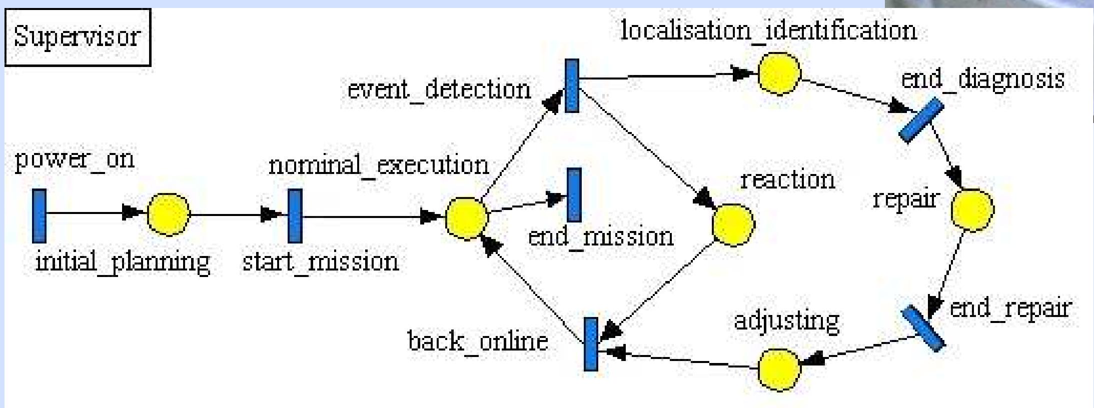
- Results
  - Successful execution
  - Use of ProCoSA for a two-robot mission





Future experiments : team operation

- Search and Rescue mission
- Partially known urban environment
- Obstacles
- 4 Pekee robots:
  - 2 UGVs
  - 2 "UAVs" (on plexiglas pane) to detect ground obstacles
- A supervisor Petri net



[5] UGVs

ONERA

Conclusions

- ProCoSA software package for decisional autonomy
  - supervises mission execution
  - deals with environmental uncertainties
  - mission data and deliberative tasks independent → modularity, genericity
  - hierarchical modelling of system behaviour

- Implementation in different autonomous vehicles
  - AUV → sea experiments
  - autonomous UAV → near future flight tests
  - autonomous UGVs → two-robot mission

Future work

- Classification of events and associated reactions, multiple event processing
- Improvement of embedded functions / real time constraints: perception, situation assessment, planning (e.g. objective planning), guidance
- Parallel research: vehicle, sensors, payload, pilot function and links with embedded functions (e.g. planning according to data quality)

- Simulation tools
- Decision Support Systems to help the operator
- Adaptive autonomy, shared control research

- Insertion of autonomous vehicles in future operational theatres
- Military and civil applications