

Architecture for Autonomy

Sara Fleury & Félix Ingrand

(sara@laas.fr felix@laas.fr)

LAAS-CNRS

(www.laas.fr)

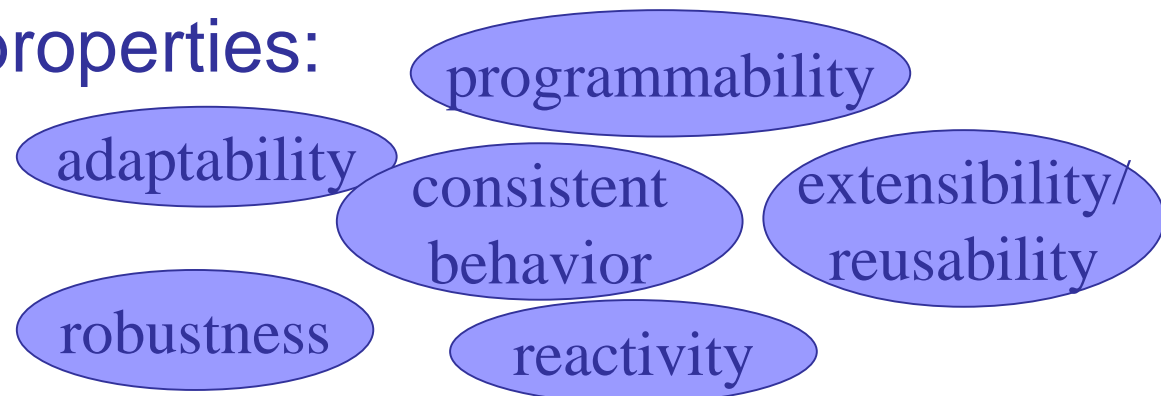
7 avenue du Colonel Roche

F-31077 Toulouse FRANCE

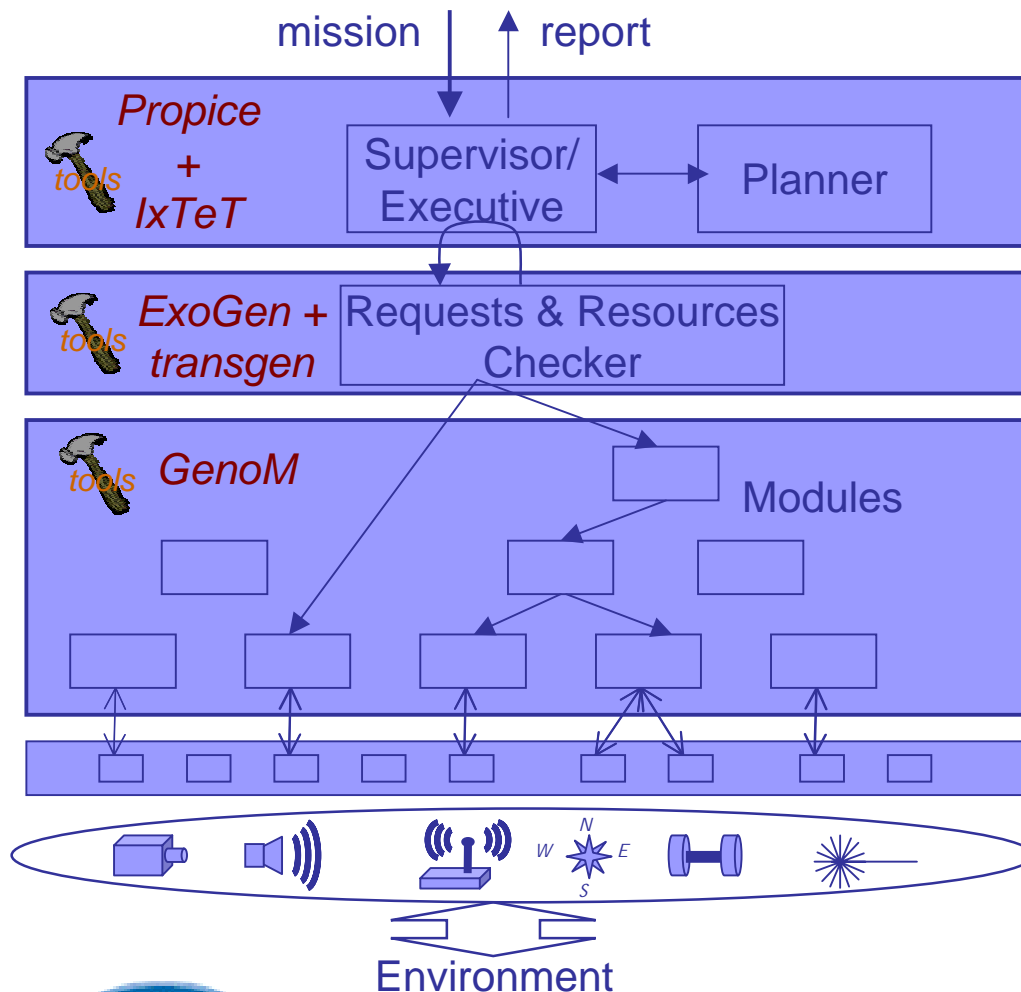


Motivations

- **Methodologies** and **tools** to design embedded software architectures for autonomous systems
- Results taken from **robotic research**
- Application to new other systems:
on board execution control and mission management
 - ground station maintenance simplified
 - flexibility and high level interactions
- Autonomy => reactive + decision making capabilities
- Architecture properties:



The 3 levels LAAS architecture: from decision to action



1. Decision Level
(planning and supervision of action)

2. Execution Control Level
(actions coordination)

3. Functional Level
(actions execution)

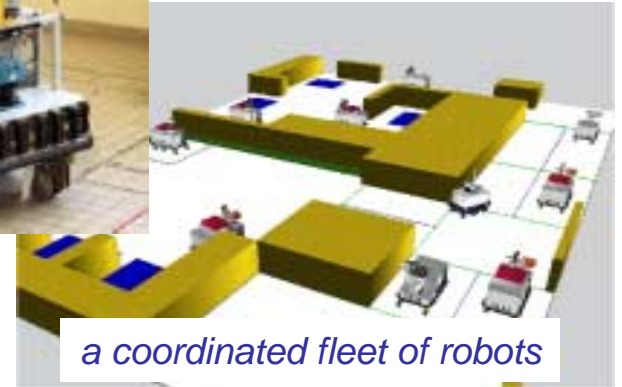
Logical System

Physical Platform

From autonomous mobile robots ...



an Hilare robot with a trailer and a 6dof arm



Service robotics

Planetary exploration robotics

Trans-shipment robotics



the rover Lama



SNCF robot Commutor



new harbour of Rotterdam

... to autonomous satellites

The screenshot displays the XPropice 2.0.0 Beta software interface, which is used for simulating autonomous satellites. The interface is divided into several windows:

- SUPERVISOR Window:** Contains a list of control actions (Actif, Oisif, Stoppé, Continuer, Pas à Pas, Arc Suivant, Arrêter, RAZ) and a log of system events. The log shows messages such as "SUPERVISOR: getting the (EVENT SYDRE START ORBCP_OUT 1) 1" and "SUPERVISOR: getting the (EVENT SYDRE START CAMERA 1) from IXTET".
- EXECUTIVE Window:** Shows the status of the executive process, including "Chargement/Compilation fichier" and "OP terminé, Fermeture fichier".
- GDHE Window:** Displays a 3D visualization of the Earth with a satellite in orbit. The satellite is shown as a small object with solar panels, positioned above the Earth's surface. The Earth is rendered in blue and white, with red dots indicating various points of interest or mission locations.
- Terminal Window (emacs@pif):** Shows the command-line interface for the simulator. It contains several commands and their outputs, including:

```
add supervisor (for-ixtet "
;; DEMARRER LA BOUCLE DE GE
add supervisor (RQST-CYCLE)

;; REQUETE UTILISATEUR
add supervisor (IMAGE-RQST (Z
6800) (IMAGE HIGH NONE NONE (

;; REINITIALISER LE PLANNER
add supervisor (for-ixtet "reset" $i)

-----Emacs: menu (Fundamental)--L12--16%-----

propice-server> add supervisor (IMAGE-RQST (ZONE TOULOUSE -33.92 1\
51.17 30 43200 46800) (IMAGE HIGH NONE NONE (..) 1001 4)

propice-server> |

***-Emacs: *propice-server* (propice-server:run)--L35--Bot----
```
- Statistics Window:** Displays performance metrics for the simulation, including:

```
----- some stats: -----
* nb developed nodes : 20
* nb backtrack points : 0
* average branching factor: 1,1 (*)
* time spent : 0,7 s
  o feasibility : 23% (*)
  o satisfiability: 43% (*)
  o resources : 12% (*)
  o propagations : 30% (*)
(*) values computed on the solution path.

Exporting solution plan to "SUPERVISOR" ... done (1) tasks
```

The bottom of the screen shows a taskbar with various application icons, including "Démarrer", "sirius...", "pif-pts...", "tinguel...", "Emacs...", "gv: tra...", "SyDR...", "XPropi...", "XProp...", "SyDR...", "Position", "GDHE", and a system clock showing "16:07".



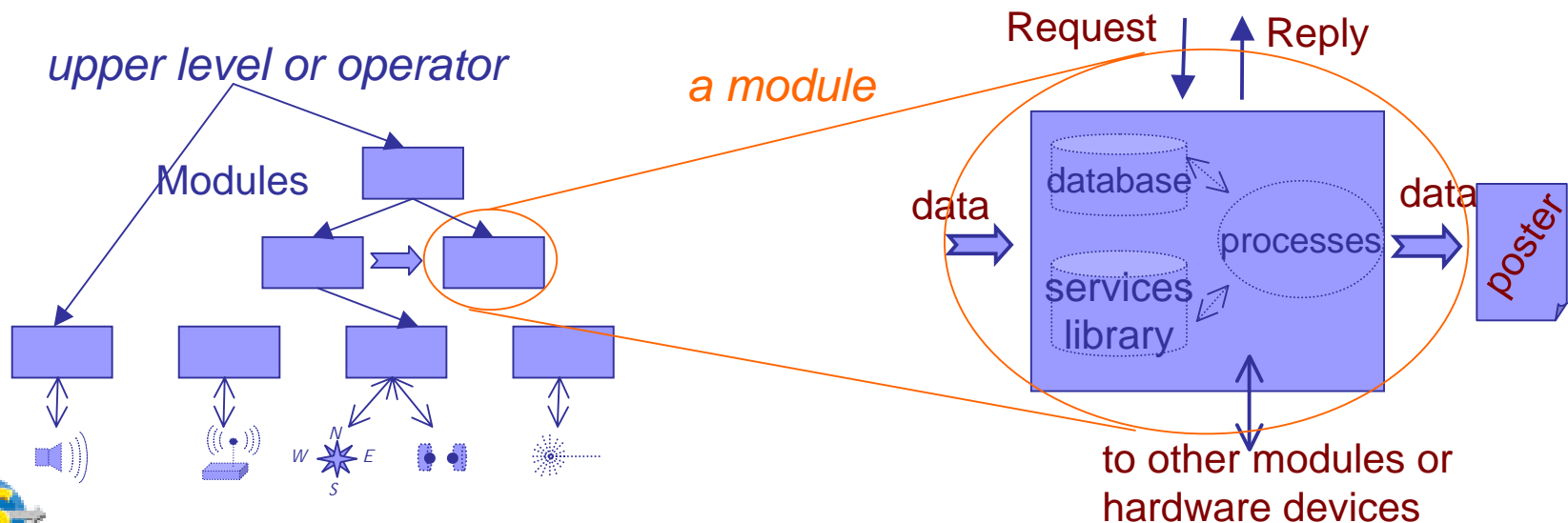
LAAS level 1: the Functional Level

Integrates all the operational functions

(hardware control, servo-control, data processing, ...)

Structured as a set of independent **modules** (dynamically controlled by the upper level)

Module: entity responsible for a physical or logical resource





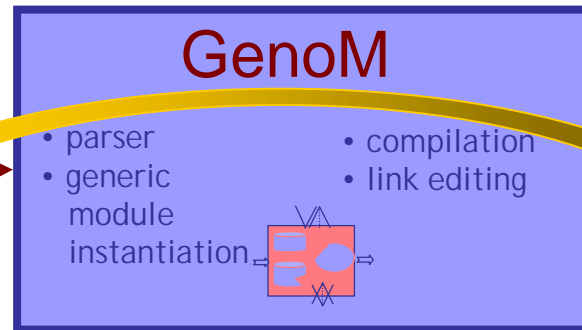
The Generator of Modules GenoM

- Automatic code synthesis
- No need to know the underlying OS
- One can concentrate on the functionalities
- Incremental design

1. module description

```
module Motion {  
  number: 9600;  
  SDI: MOTION_DATA;  
}  
request SetPos {  
  type: control;  
  input: pos::pos;  
  control: controlPC;  
  report: BAD_PARAM;  
}  
task Move {  
  period: 25;  
  priority: 15;  
}
```

2. module generation



*executable module
(various OS)*

*Interface libraries
(C, Propice, TCL, ...)*


test programs

3. algorithms integration

```
STATUS  
controlPos (POS_STR *pos_REPORT +  
{  
  if (*pos == 0) {  
    *report = BAD_PARAM;  
    return ERROR; }  
  return OK;  
}
```

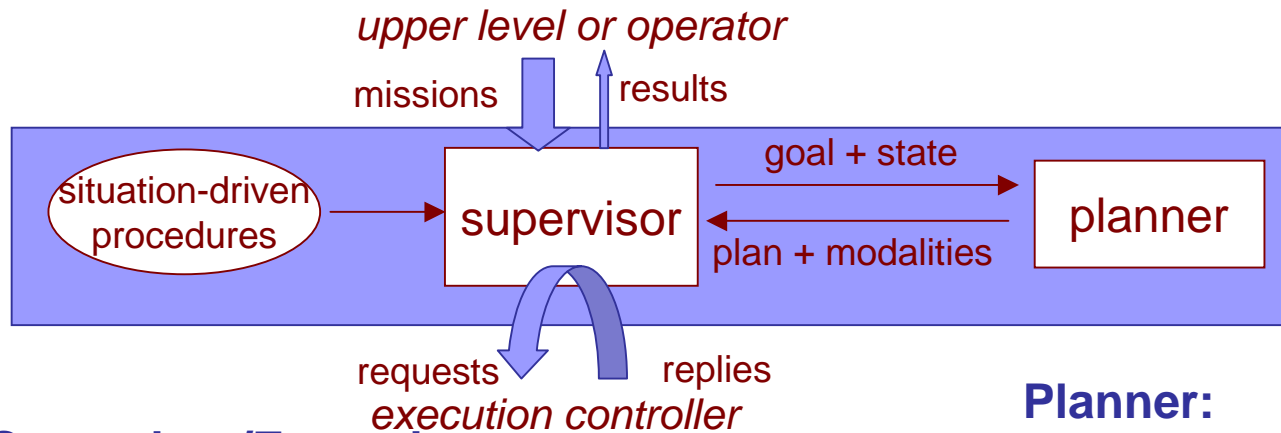
4. tests

LAAS level 2: the Execution Control level


- **Pivot** between functional/decision levels
- Purely **reactive system** that reacts to decision level requests and functional level replies
- **State controller** of function level:
 - maintains functional level state
 - filters decision level requests
 - detects and manages conflicts
 - check resources usage
-  **Transgen/Exogen** : automatic **BDD** from a set of constraints/rules (consistent, optimised)
- Offline model checking with temporal logic

LAAS level 3: the Decision Level


- All processes that require anticipation and global knowledge of the task and of the execution context.
- Structured in **supervisor/executive-planner** layers:



Supervisor/Executive:

- Interprets upper mission
- Selects action procedures (or call planner)
- Controls the procedures execution
- Reacts to events (replies) from lower level
-  : PROPICE

Planner:

- Queried by supervisor
- Deals with:
 - time constraints
 - resources constraints
 - predictable events
- Produces plan of actions
-  : IxTeT



The Procedural Reasoning System PROPICE

Properties:

- high-level language
- parallel tasks + asynchronous events handling
- temporal properties

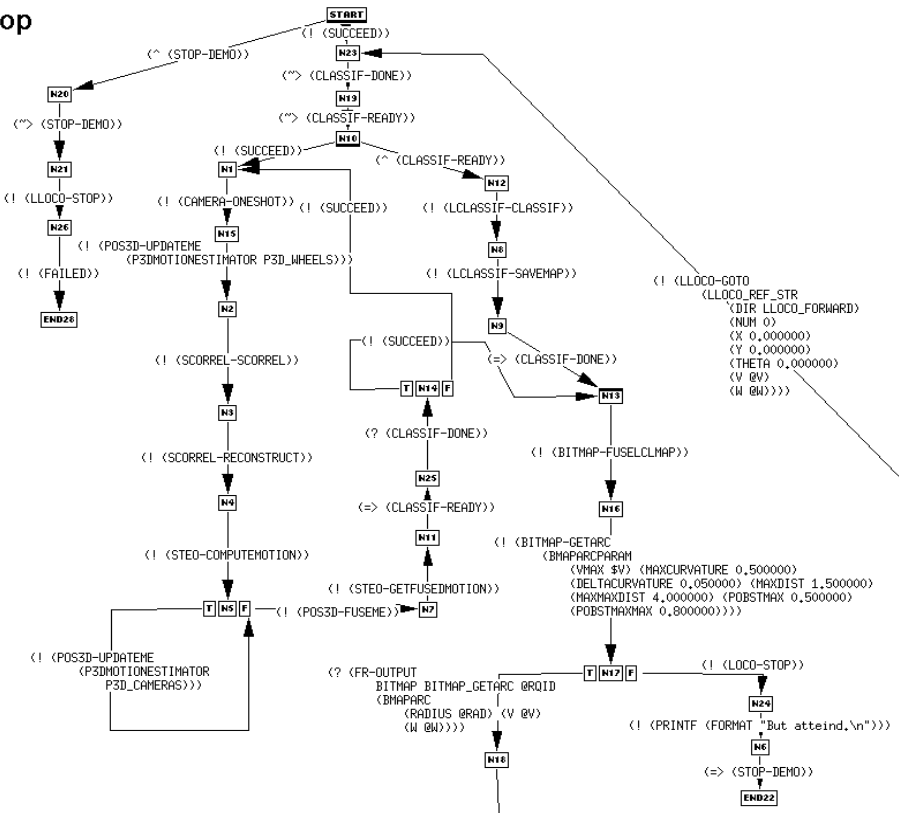
Main components:

- automatically updated **database** (view of the world)
- a library of **procedures**:
 - sequence of actions and tests
 - to achieve given goals, or
 - to react to certain situations
- a dynamic task graph

Manip Loop

INVOCATION:
<I MANIP-LOOP>

CONTEXT:
<VITESSE \$V>



Example of a PROPICE procedure



IxTeT Temporal Planner

IxTeT: IndeXed Time Table

- IxTeT kernel: an efficient time-map manager
- Time-point algebra relations and restricted interval algebra
- Used in situation recognition and plan synthesis
- Common knowledge representation: **chronicles**

Example of an IxTeT plan

