

Robots

Mobiles & Autonomes

avec

Pharo

Noury Bouraqadi – JM2L 2010

Noury Bouraqadi

<http://car.mines-douai.fr/noury>

JM2L – 27 nov 2010

Robots

Mobiles & Autonomes

avec

Pharo

Robotique Industrielle



Nouraqadi – JM2L 2010

*Première
Application*

Robots?

1. Machine, automate à l'aspect humain, capable de se **mouvoir et d'agir**. => androïde, humanoïde [...]
2. Mécanisme **automatique** à commande électronique pouvant se substituer à l'homme **pour effectuer certaines opérations**, et capable d'en modifier de lui-même le cycle en **appréhendant son environnement** (=>automatique, cybernétique) [...]

Le petit Robert

Robots?

Mobilité

1. Mécanisme, automate à l'aspect humain capable de se **mouvoir et d'agir**. => autonome [...]

Autonomie

2. Mécanisme **automatique** à commande électronique pouvant se substituer à l'homme **pour effectuer certaines opérations**, et capable d'en modifier de lui-même le cycle en **appréhendant son environnement** (=>automatique, cybernétique) [...]

Le petit Robert

Robots

Mobiles & Autonomes

avec

Pharo

Robot Mobile

- **Environnement** partiellement connu et changeant
 - Carte ? Lois Physiques ? Dynamique ?
- **Problème de navigation**
 - Localisation
 - Reconnaissance
 - Chemins
 - Obstacles

Robot Autonome

Décide seul

des

actions à réaliser

en

fonction de ses **perceptions**

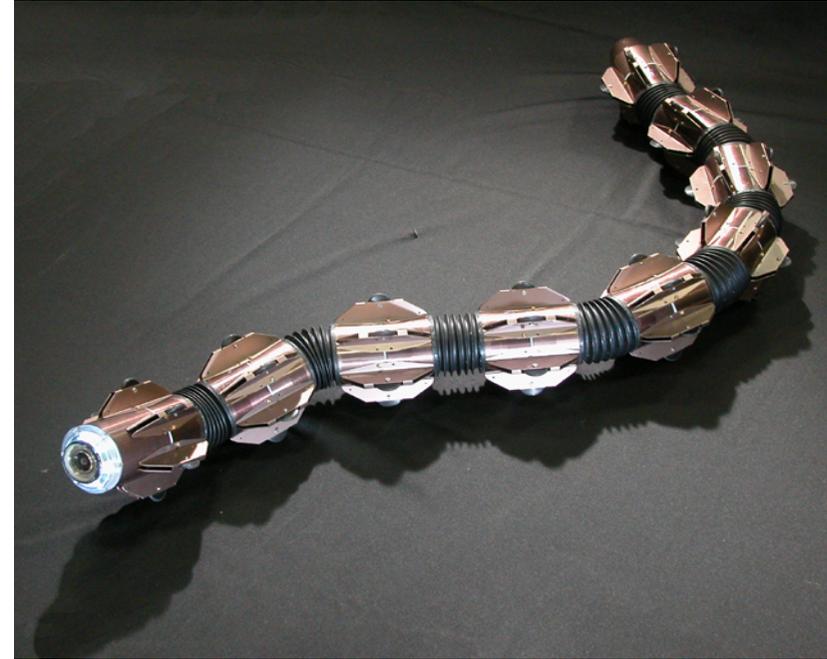
Robotique Industrielle



Noury Bouraqadi – JM2L 2010

- Fonctions/trajectoires pré-définies + figées
- Pas d'interaction avec l'humain
- Environnement maîtrisé/connu

Robots Mobiles & Autonomes



Robots Mobiles & Autonomes

2L 2010



Robots Humanoïdes

Robots Mobiles & Autonomes

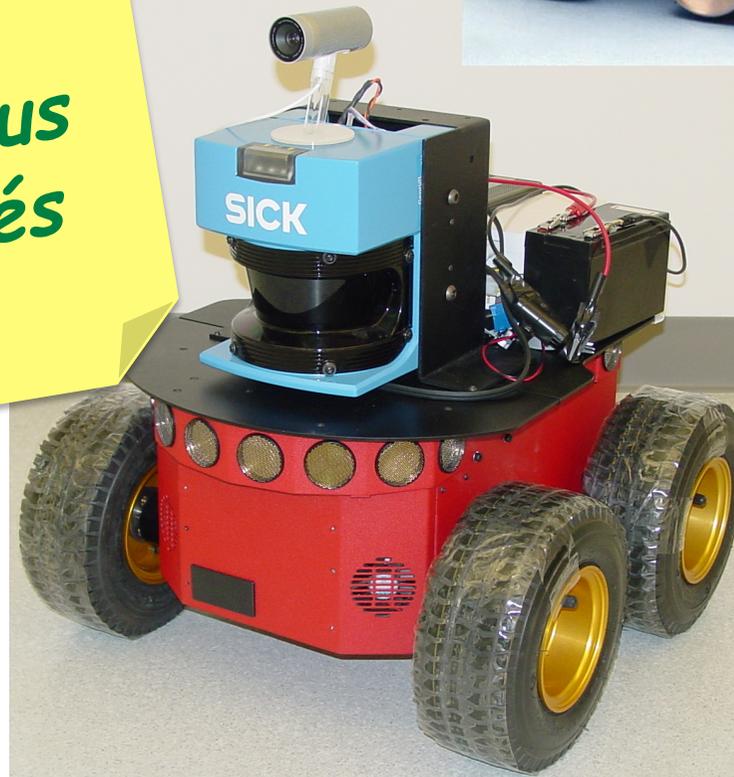


Robots Androïdes

Robots Mobiles & Autonomes



Les plus
utilisés



Défis

Noury Bouraqadi – JM2L 2010

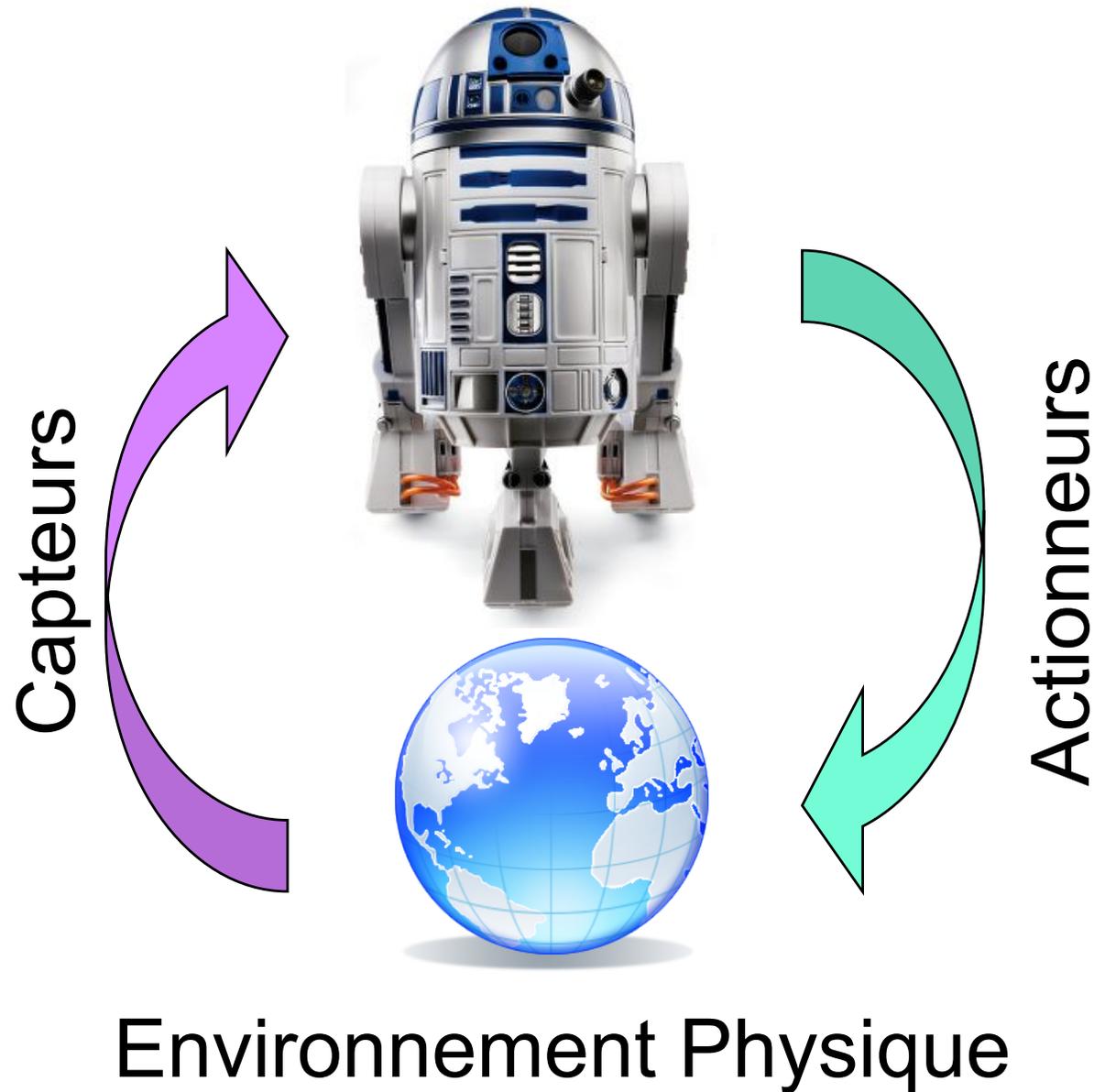
*Ressources
Limitées*

*Robots
Rapides*

*Robots
Intelligents*



Logiciel de contrôle d'un robot



Logiciel de contrôle d'un robot

- Sense

- Lire les capteurs
- Extraire des informations utiles

- Plan

- Prendre des décisions

- Act

- Exécuter des actions simples
- Ex: Avancer, reculer, tourner, ...

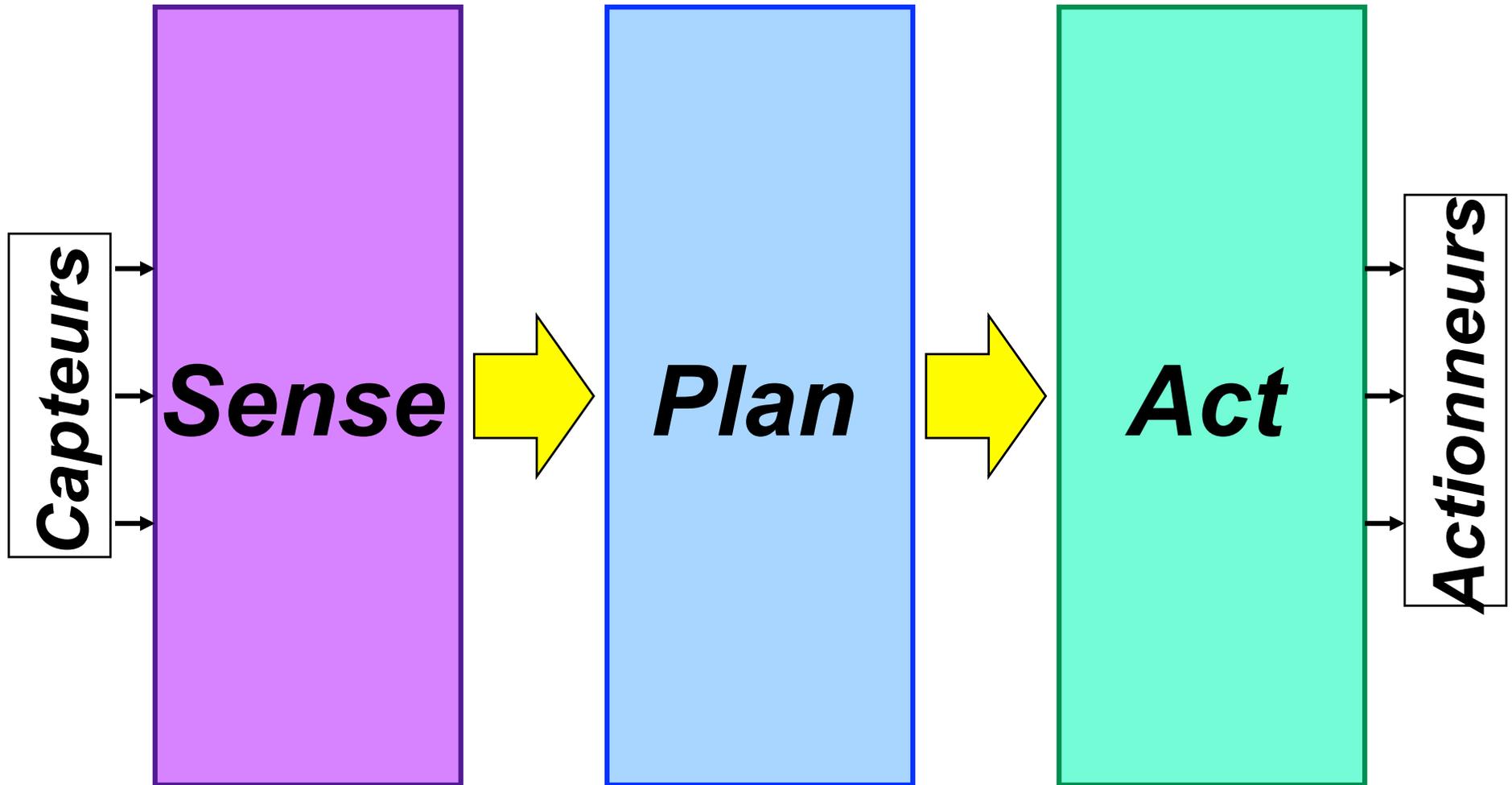
Familles d'architectures

- Délibératives
- Réactives
- Comportementales
- Hybrides

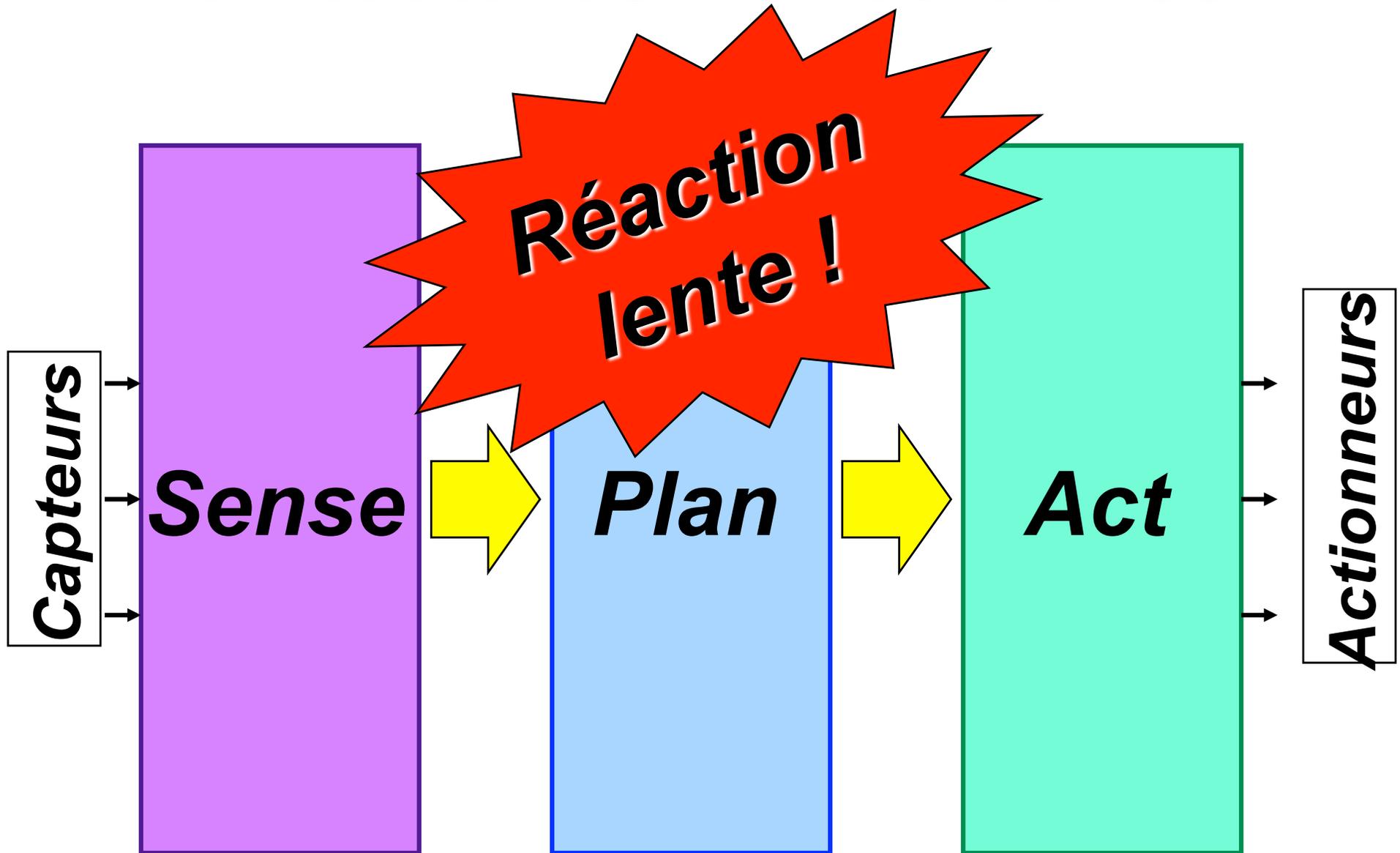
Architectures Délibératives

Think then Act

Noury Bouraqadi – JM2L 2010



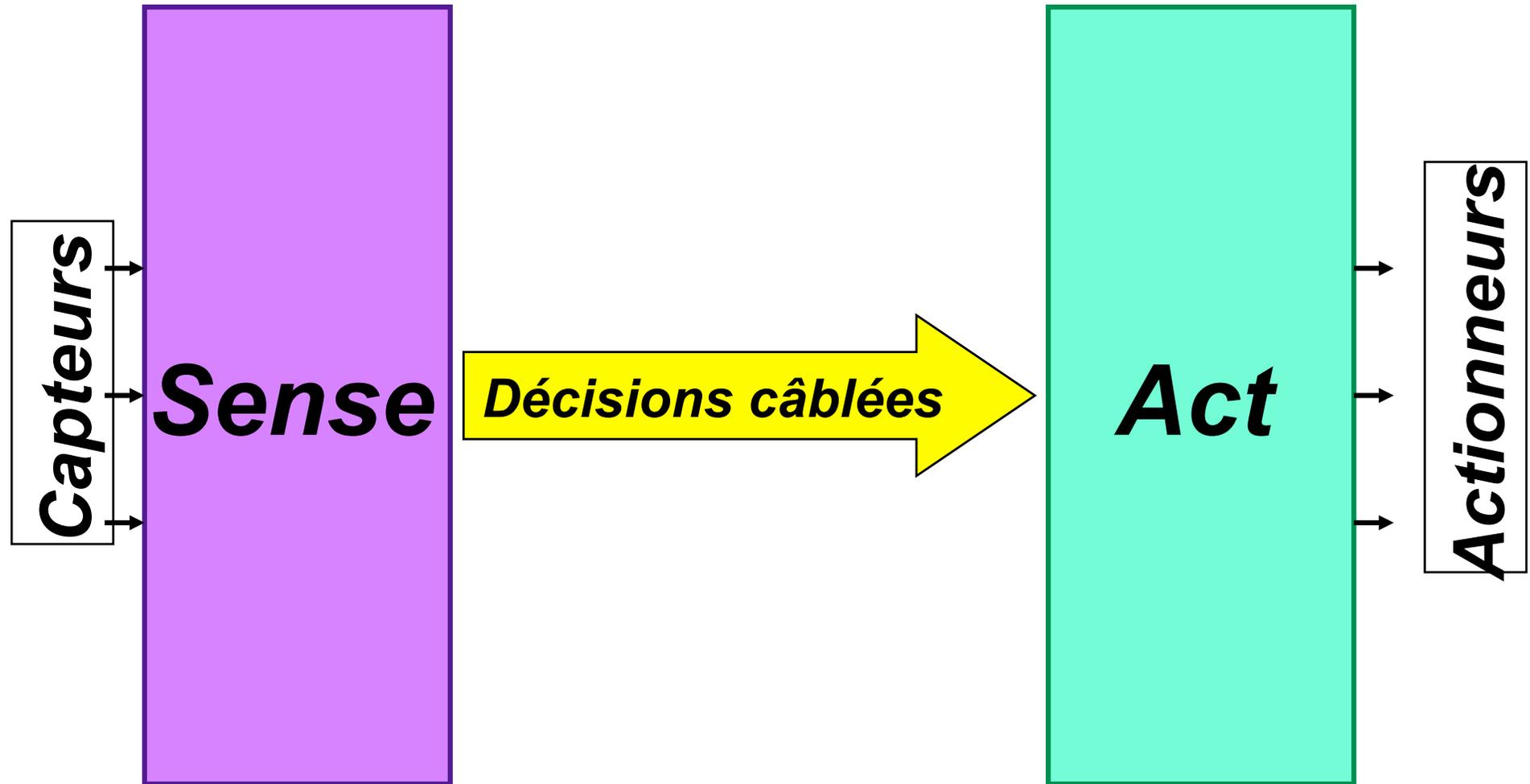
Architectures Délibératives



Architectures Réactives

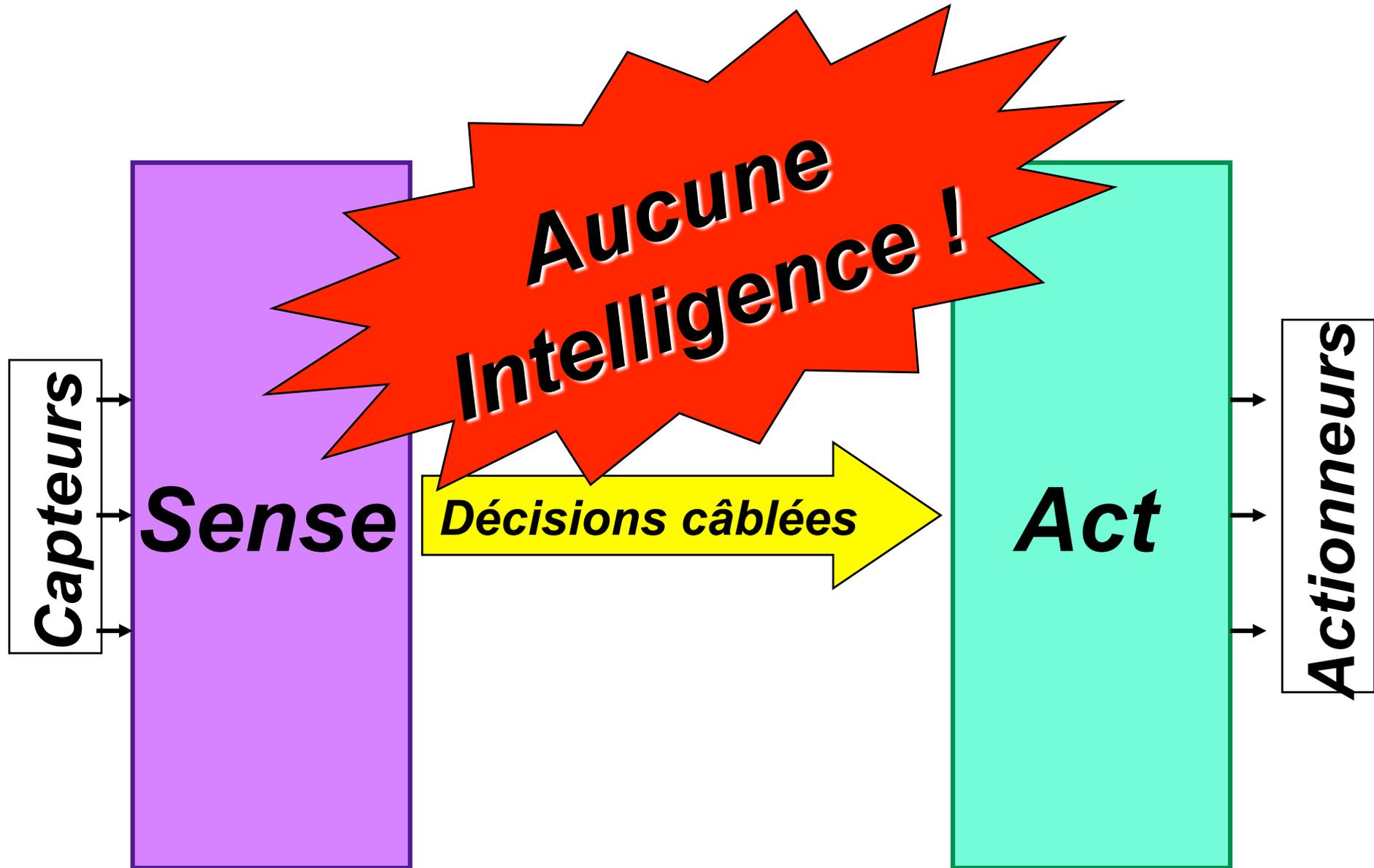
Don't think, (Re)Act

Noury Bouraqadi - DIA - EM Douai



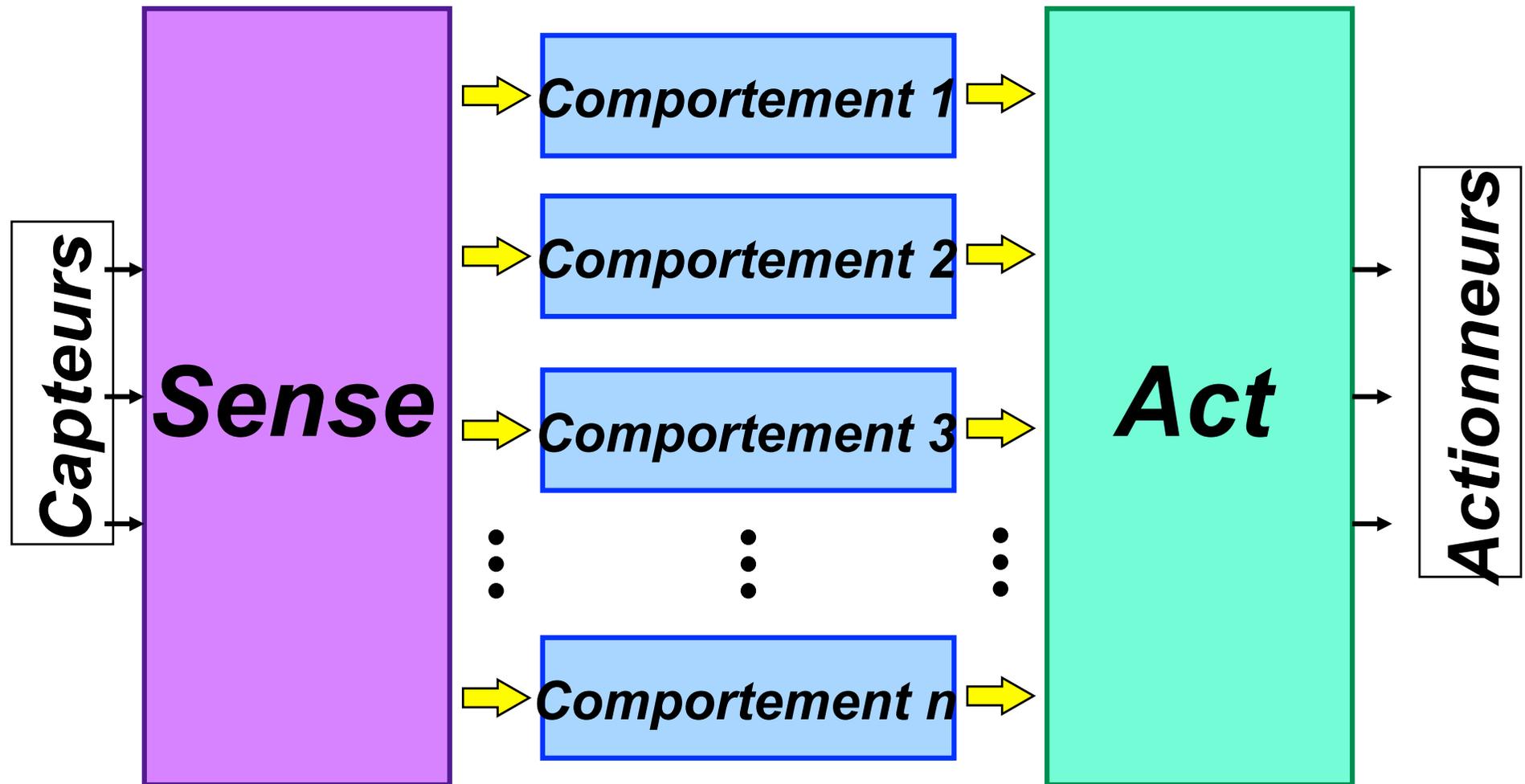
Architectures Réactives

Noury Bouraqadi – JM2L 2010



Architectures Comportementales

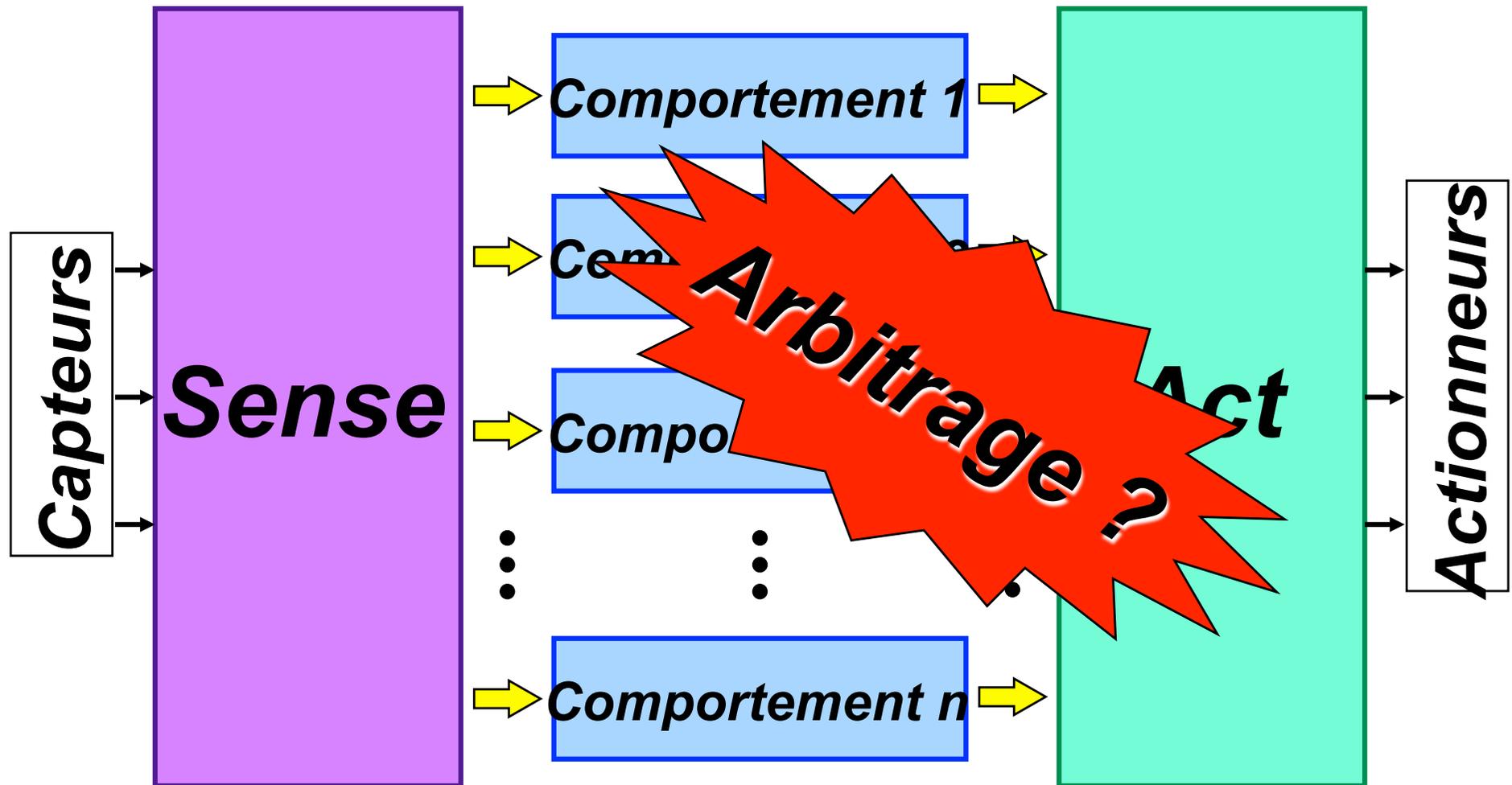
Think the Way You Act



Architectures Comportementales

Think the Way You Act

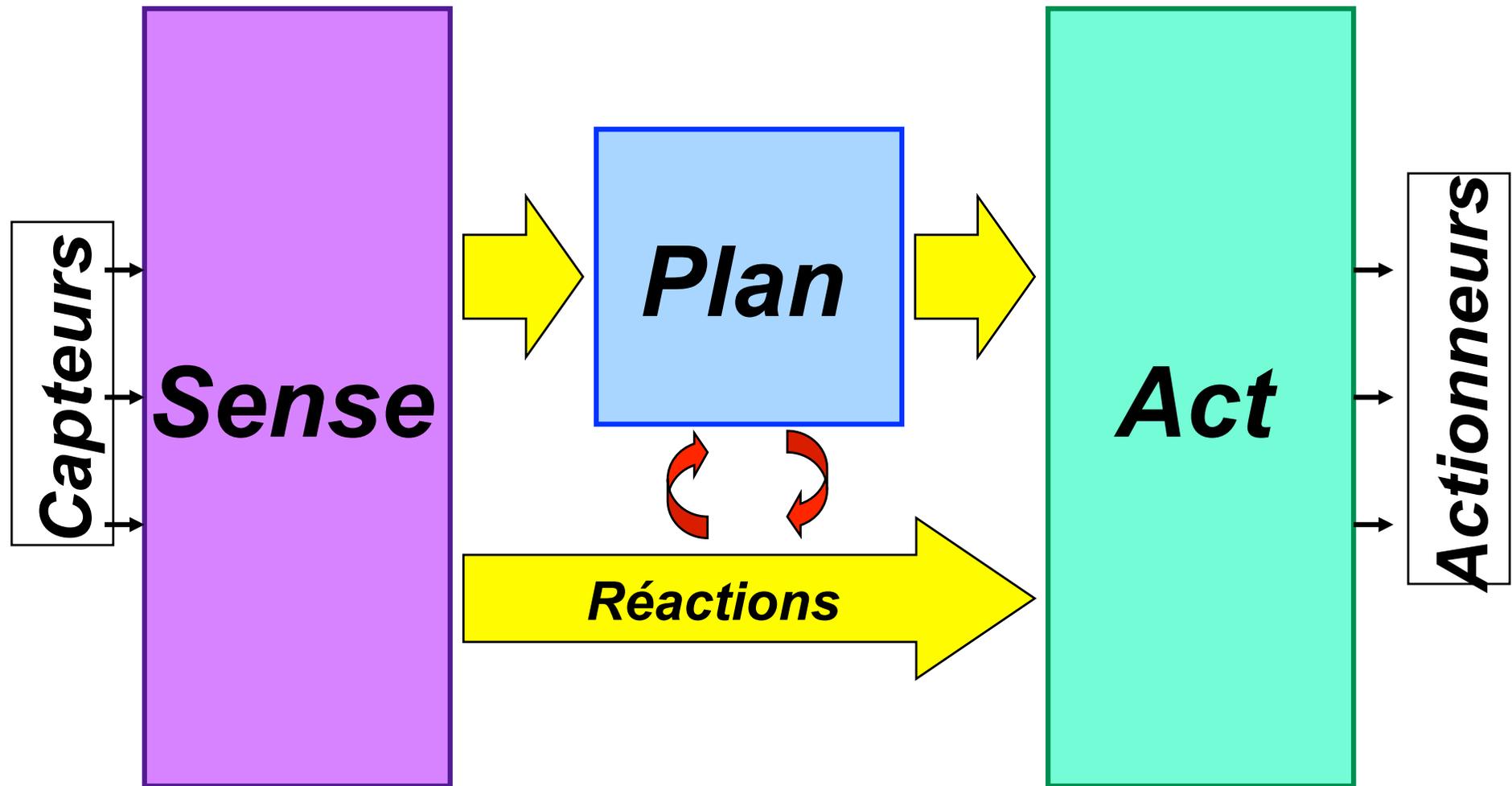
Noury Bouraqadi – JM2L 2010



Architectures Hybrides

Think and Act Concurrently

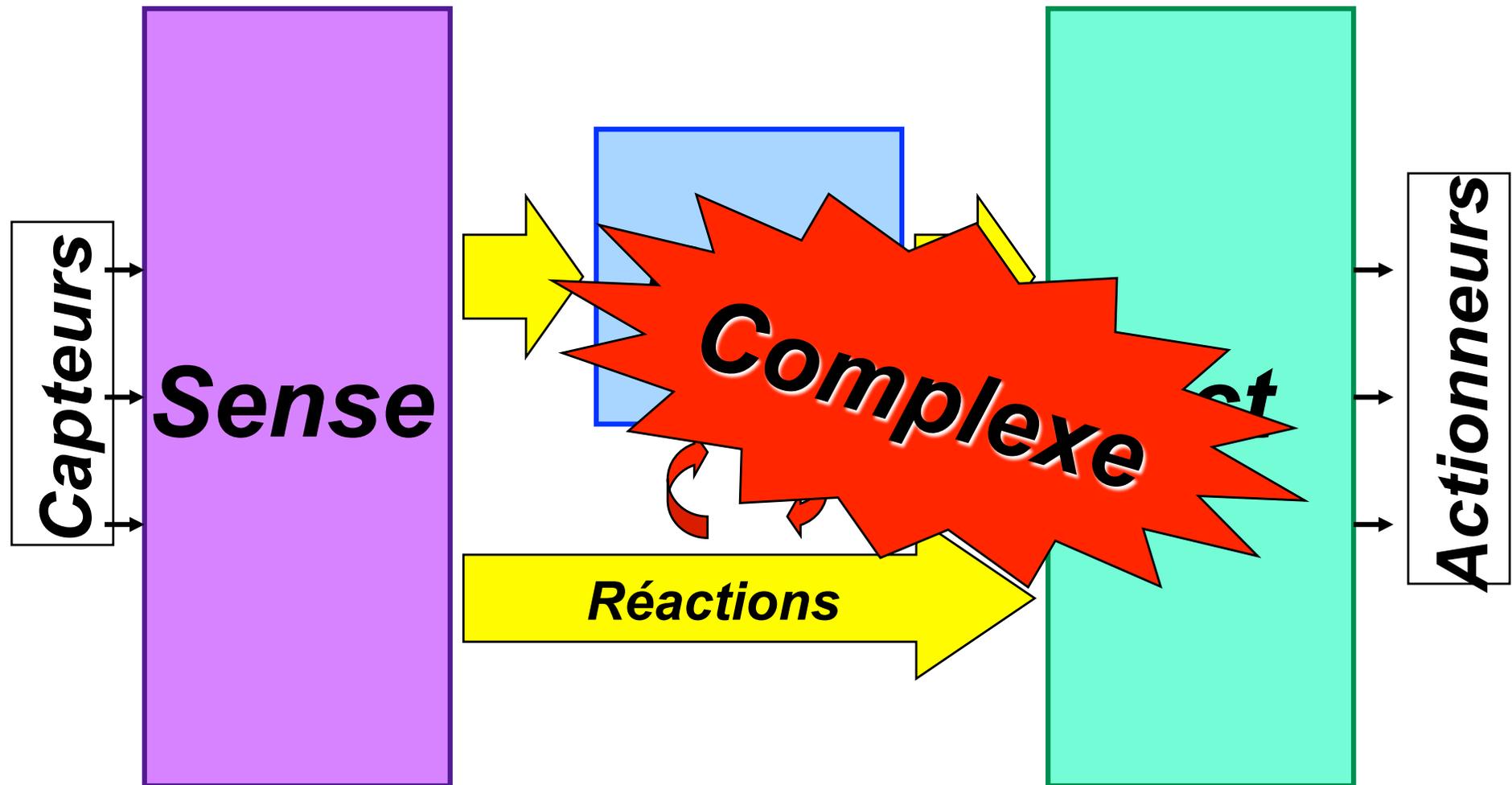
Noury Bouraqadi – JM2L 2010



Architectures Hybrides

Think and Act Concurrently

Noury Bouraqadi – JM2L 2010



Robots

Mobiles & Autonomes

avec

Pharo



*is a
Clean
Innovative
Open-Source
Smalltalk-inspired
Environment*



is a

Clean

Innovative

Open-Source

malltalk-inspired

Environment

Noury P. ... JM2L 2010

Langage
+
Bibliothèques
+
Outils



is a

Clean

Innovative

Open-Source

Smalltalk-inspired

Environment

*Langage
dynamique
à objets*



is a

Clean

Innovative

Open-Source

MIT

Smalltalk-inspired

Environment

Libre !

PharO 

is a

Clean

Innovative

Open-Source

Smalltalk-inspired

Environment

Traits

*Relation
avec la
recherche*



*is a
Clean*

*Finalité
Industrielle
&
Pédagogique*

*Innovative
Open-Source
Smalltalk-inspired
Environment*

Robots

Mobiles & Autonomes

avec

Pharo

par l'exemple

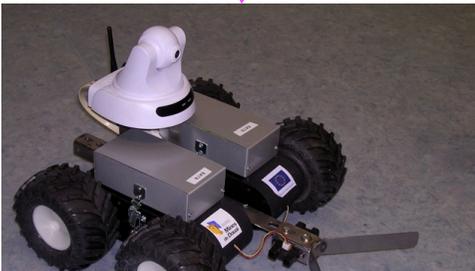
ILLUSTRATION

avec

WifiBotST

Framework

pour programmer



Robots Physiques

ILLUSTRATION

avec

WifiBotST

Développé en



Framework

pour programmer



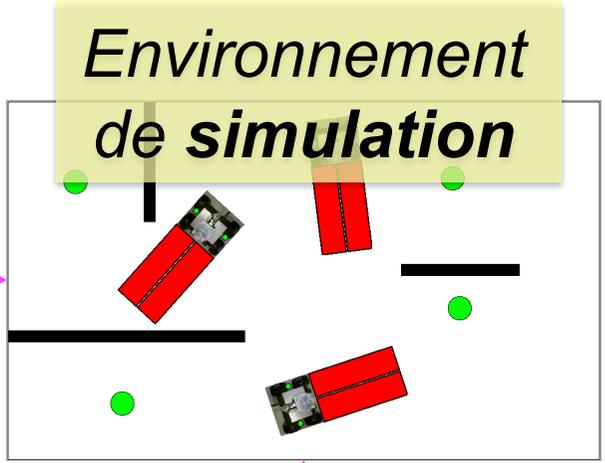
Robots Physiques

ILLUSTRATION

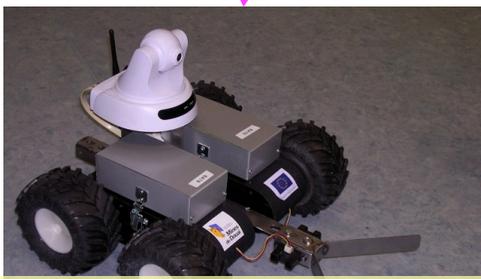
avec

WifiBotST

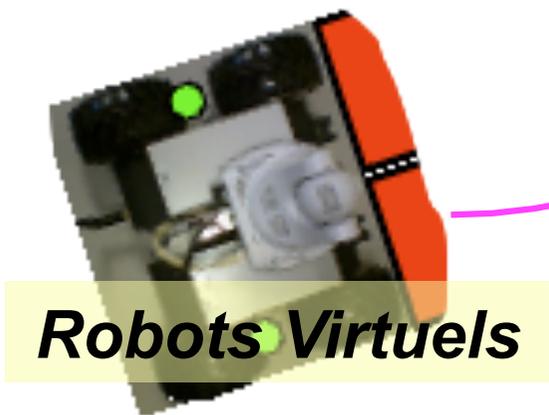
Développé en
Pharo



Framework
pour programmer



Robots Physiques

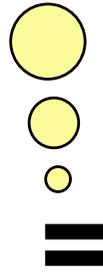


Robots Virtuels

Noury Bouraqadi – JM2L 2010

*Oubliez l'archaïsme
des fichiers !*

Application



**Ensemble
d'objets**

Naviguer/Editer le code

The screenshot displays the Robot IDE interface. The window title is "Robot". The interface is divided into several sections:

- Left Panel (Class List):** A list of classes including ToolsTest-Inspector, ToolsTest-MessageTally, Traits-Composition, Traits-Kernel, Traits-Kernel-Traits, TrueType-Fonts, TrueType-Support, VB-Regex, VB-Regex-Exceptions, RoboST-Morphic GUI, RoboSTTest-MorphicGUI, and Demo-JM2L. A red arrow labeled "1" points to "Demo-JM2L".
- Center Panel (Class Hierarchy):** Shows the "Robot" class selected. A red arrow labeled "2" points to the "Robot" class name.
- Right Panel (Code Editor):** Displays the code for the "Robot" class, including methods like `body`, `body:`, `defaultSpeed`, `goForward`, `initialize`, `isStopped`, `move`, `rotationSpeed`, `speed`, `speed:`, `turnLeft`, and `turnRight`.
- Bottom Panel (Navigation):** Contains buttons for "browse", "hierarchy", "variables", "implementors", "inheritance", "senders", "versions", and "view".
- Bottom Panel (Object Information):** A red-bordered box contains the following information:

```
Object subclass: #Robot
instanceVariableNames: 'body speed'
classVariableNames: ''
poolDictionaries: ''
category: 'Demo-JM2L'
```

A red arrow labeled "3" points to this box.

Naviguer/Editer le code

The image shows a screenshot of a development environment window titled "Robot". The window is divided into several panes and has a toolbar at the bottom. Red arrows and numbered circles (1-5) indicate a sequence of navigation steps:

- 1**: Points to the "Demo-JM2L" package in the left-hand package browser.
- 2**: Points to the "Robot" class in the middle pane.
- 3**: Points to the "moving" method in the right-hand method list.
- 4**: Points to the "goForward" method in the right-hand method list.
- 5**: Points to the code snippet `self body goForwardBy: self speed` in the bottom pane, which is enclosed in a red box.

The toolbar at the bottom includes buttons for "browse", "hierarchy", "variables", "implementors", "inheritance", "senders", "versions", and "view".

Démo 1

Interface avec l'OS = 4 fichiers

- Machine virtuelle
- **Image mémoire** : lecture / écriture
- Source des bibliothèques de base : lecture seule
- Sources des changements :
 - lecture / **sauvegarde automatique**
 - Quasi-impossible de perdre les sources !
 - **Versionning** sans effort !



Langage *Simple*

- Peu de concepts / règles
 - 6 Mots réservés
 - 13 caractères spéciaux
 - 5 concepts centraux
 - 4 expressions de base
 - 1 mode de gestion de la mémoire

Langage *Simple*

- Peu de concepts / règles

- 6 Mots réservés

- 13 caractères

- 5 conditions

- 4 expressions

- 1 mode

- **nil** objet indéfini
 - valeur par défaut des variables
- **true, false** objets booléens
- **self** l'objet courant
- **super** l'objet courant dans le contexte de la superclasse
- **thisContext** partie de la pile d'exécution correspondant à la méthode courante

Langage *Simple*

- Peu de concepts / règles
 - 6 Mots réservés
 - 13 caractères spéciaux

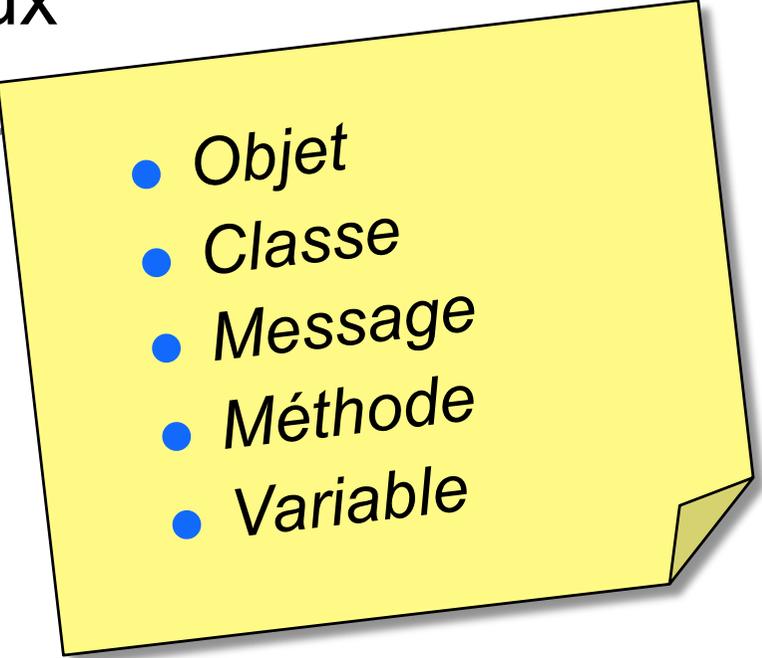
- *:=* affectation
- *^* return
- *| variable1 variable2 |*
- *\$* caractère
- *#* littéral
- *{ }* tableau

- *.* termine toute expression
- *;* cascade de messages
- *()* précedence
- *[]* bloc de code
- *"* commentaire *"*
- *<pragma>*
- *'*chaîne de caractères *'*



Langage *Simple*

- Peu de concepts / règles
 - 6 Mots réservés
 - 13 caractères spéciaux
 - 5 concepts centraux
 - 4 expressions de base
 - 1 mode de gestion

- 
- Objet
 - Classe
 - Message
 - Méthode
 - Variable



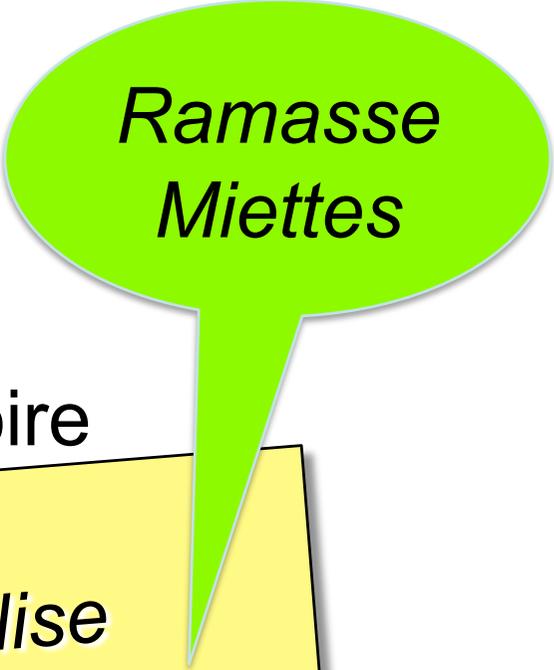
Langage *Simple*

- Peu de concepts / règles
 - 6 Mots réservés
 - 13 caractères spéciaux
 - 5 concepts centraux
 - 4 expressions de base
 - 1 mode de gesti

- *Déclaration d'une variable*
- *Affectation d'une variable*
- *Lecture d'une variable*
- *Envoi d'un message*

Langage *Simple*

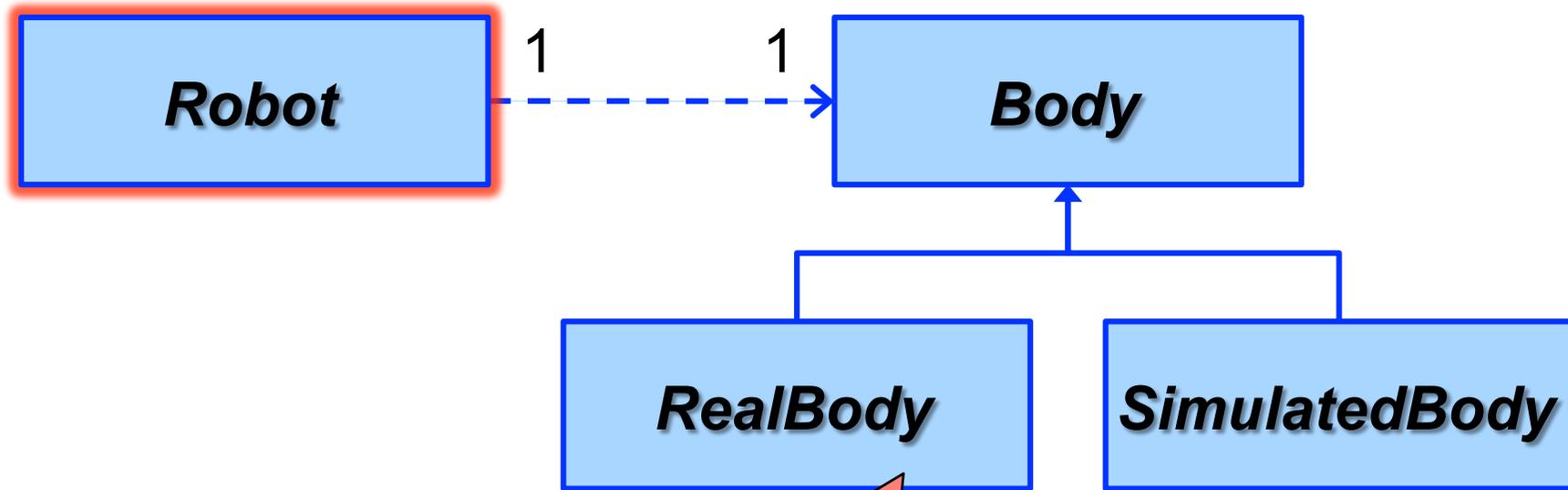
- Peu de concepts / règles
 - 6 Mots réservés
 - 13 caractères spéciaux
 - 5 concepts centraux
 - 4 expressions de base
 - 1 mode de gestion de la mémoire



*Ramasse
Miettes*



Automatique utilise
un "Garbage Collector"

**Démo 2**



Langage *Dynamique*

Tout se passe à **l'exécution**

- Définition des classes
- Compilation
- Vérification de type
- ...



Langage *Dynamique*

Tout se passe à **l'exécution**

- Définition des classes
- Compilation
- Vérification de type
- ...



Langage *Dynamique*

Tout se passe à **l'exécution**

- Définition des classes
- Compilation
- Vérification de type
- ...



Langage *Dynamique*

Tout se passe à **l'exécution**

- Définition des classes
- Compilation
- Vérification de type
- ! ...

Langage dynamique !

Modification de la classe à l'exécution

Après la création d'instances

Noury Bouraqadi – JM2L 2010

<i>Robot</i>
speed body
speed speed: turnBy: goForwardBy: move



Démo 3



Langage ***Uniforme***

- Pas d'exception aux règles
 - **Tout** est objet
 - nombres, tableaux, classes, compilateur, IDE, ...
 - **Toute** méthode retourne une valeur
 - **Toute** action == envoi de message
 - création d'objets, définition de classes, if, while, ...
 - **Toutes** les classes et les méthodes sont publics
 - **Tous** les champs sont privés
 - **Tout** objet est instance d'une classe



Langage ***Uniforme***

- Pas d'exception aux règles
 - **Tout** est objet
 - nombres, tableaux, classes, compilateur, IDE, ...
 - **Toute** méthode retourne une valeur
 - **Toute** action == envoi de message
 - création d'objets, définition de classes, if, while, ...
 - **Toutes** les classes et les méthodes sont publics
 - **Tous** les champs sont privés
 - **Tout** objet est instance d'une classe

Itérations & Blocs de code

- Répéter un nombre de fois
- Répéter tant que





Langage ***Uniforme***

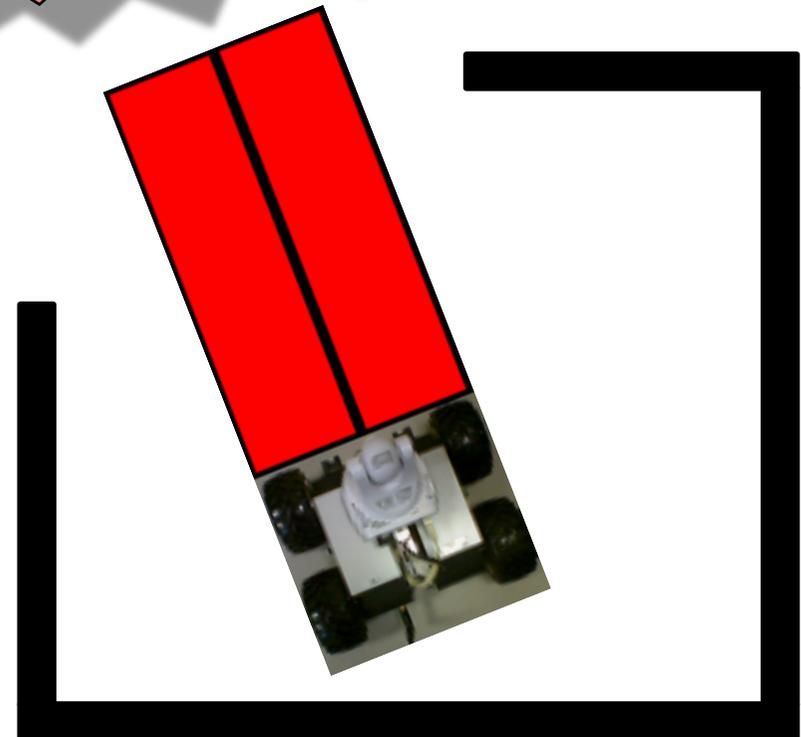
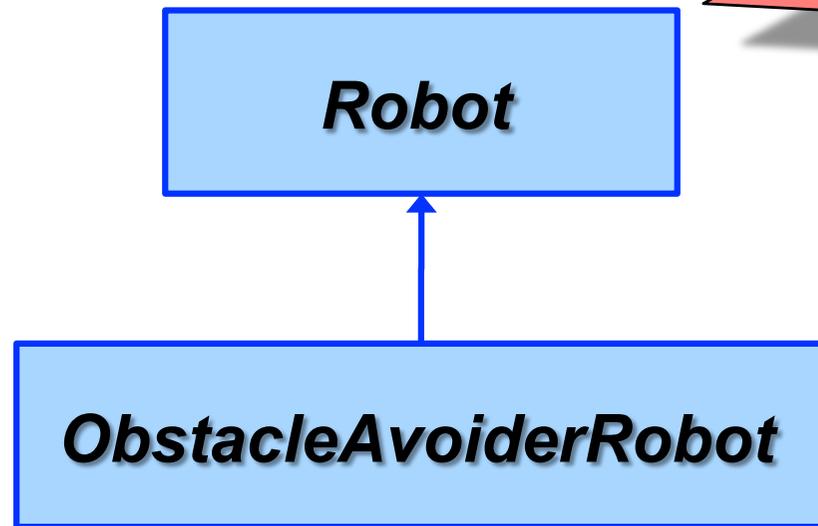
- Pas d'exception aux règles
 - **Tout** est objet
 - nombres, tableaux, classes, compilateur, IDE, ...
 - **Toute** méthode retourne une valeur
 - **Toute** action == envoi de message
 - création d'objets, définition de classes, if, while, ...
 - **Toutes** les classes et les méthodes sont publics
 - **Tous** les champs sont privés
 - **Tout** objet est instance d'une classe



Langage ***Uniforme***

- Pas d'exception aux règles
 - **Tout** est objet
 - nombres, tableaux, classes, compilateur, IDE, ...
 - **Toute** méthode retourne une valeur
 - **Toute** action == envoi de message
 - création d'objets, définition de classes, if, while, ...
 - **Toutes** les classes et les méthodes sont publics
 - **Tous** les champs sont privés
 - **Tout** objet est instance d'une classe

Héritage & Conditionnelles





Langage ***Uniforme***

- Pas d'exception aux règles
 - **Tout** est objet
 - nombres, tableaux, classes, compilateur, IDE, ...
 - **Toute** méthode retourne une valeur
 - **Toute** action == envoi de message
 - création d'objets, définition de classes, if, while, ...
 - **Toutes** les classes et les méthodes sont publics
 - **Tous** les champs sont privés
 - **Tout** objet est instance d'une classe



Langage *Uniforme*

- Pas d'exception aux règles
 - **Tout** est objet
 - nombres, tableaux, chaînes, ...
 - **Toute** méthode est publique
 - **Toute** action = méthode
 - création d'objets, accès, if, while, ...
 - **Toutes** les classes et les méthodes sont publics
 - **Tous** les champs sont privés
 - **Tout** objet est instance d'une classe

**Variables
accessibles par
un seul objet**

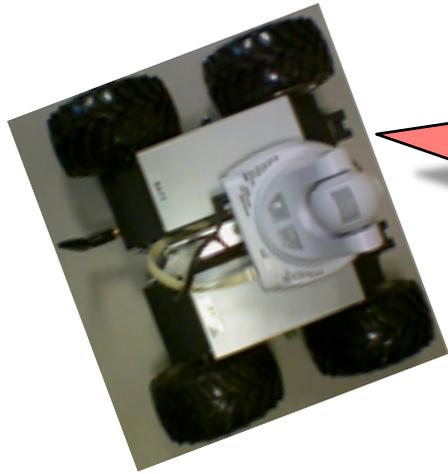


Langage ***Uniforme***

- Pas d'exception aux règles
 - **Tout** est objet
 - nombres, tableaux, classes, compilateur, IDE, ...
 - **Toute** méthode retourne une valeur
 - **Toute** action == envoi de message
 - création d'objets, définition de classes, if, while, ...
 - **Toutes** les classes et les méthodes sont publics
 - **Tous** les champs sont privés
 - **Tout** objet est instance d'une classe

Les classes sont des objets !

Noury Bouraqadi – JM2L 2010

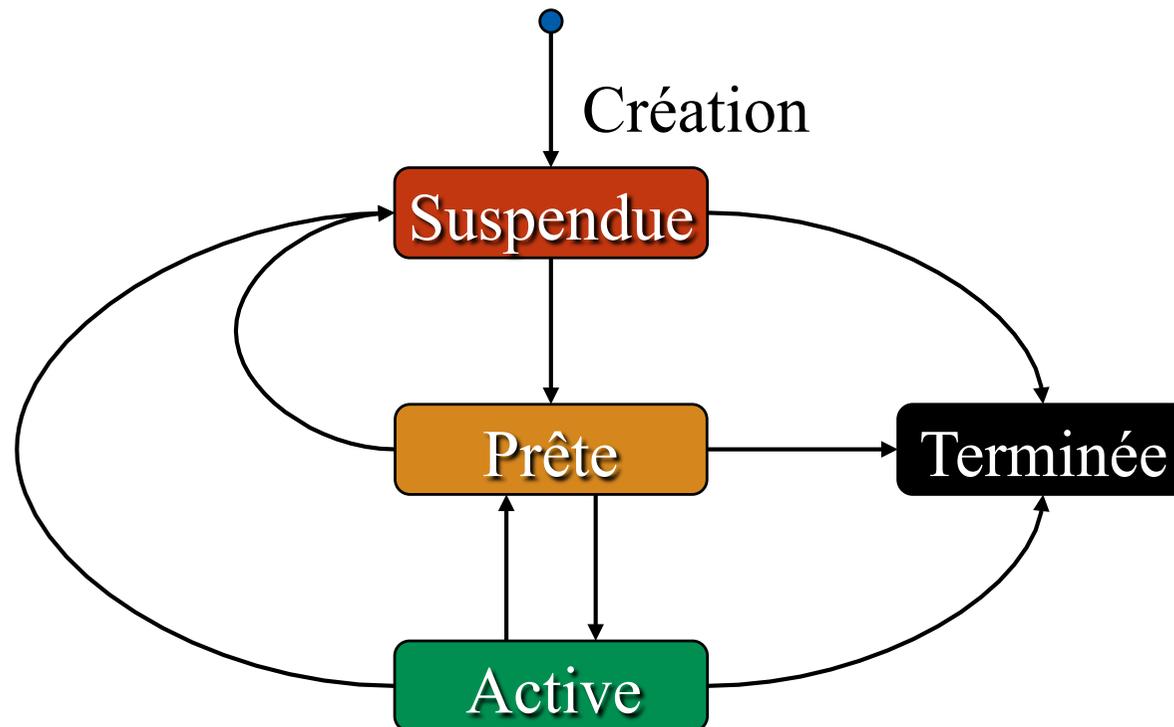


Démo 6

Les processus sont des objets

Multi-tâche préemptif

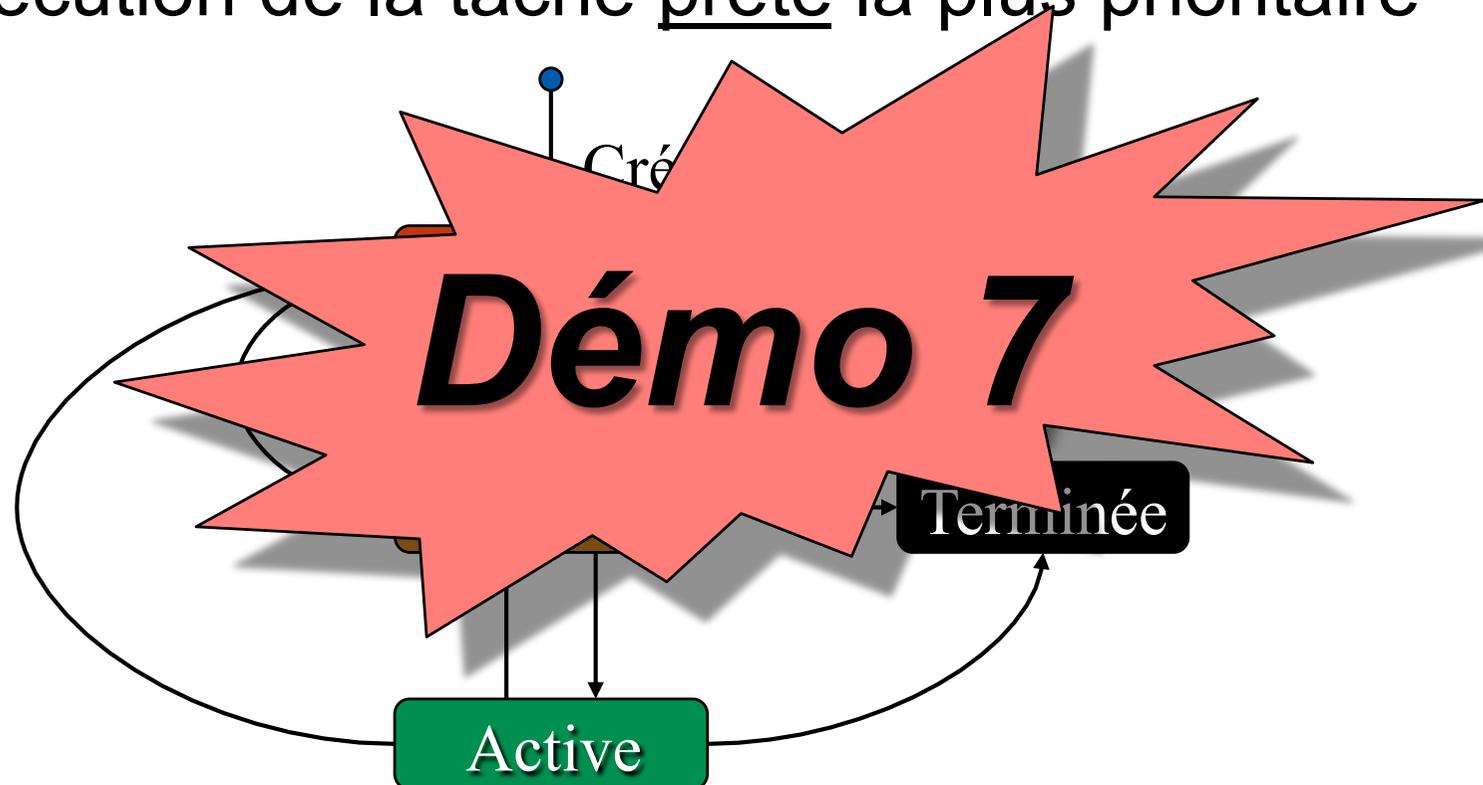
- Chaque tâche a une priorité **modifiable**
- Exécution de la tâche prête la plus prioritaire



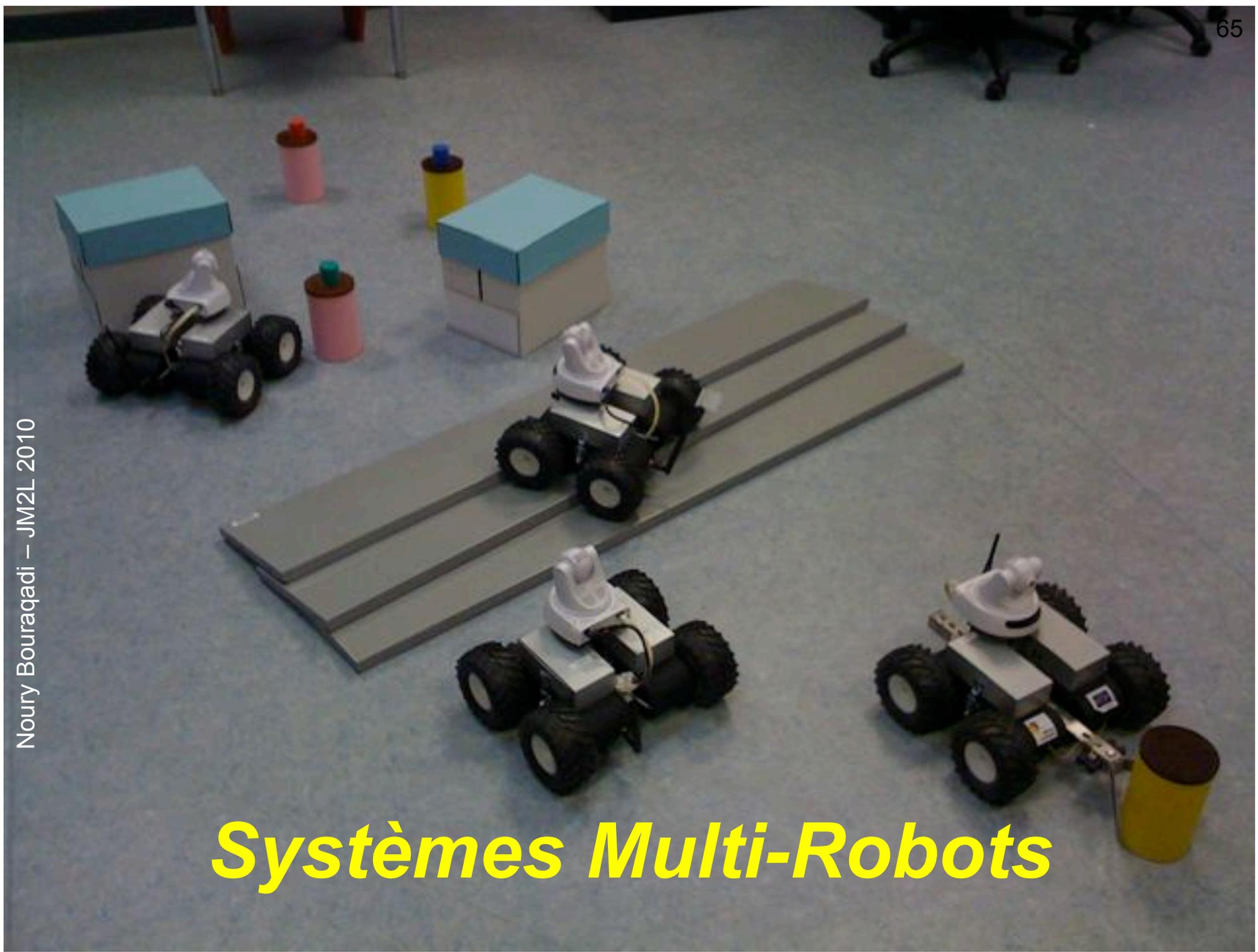
Les processus sont des objets

Multi-tâche préemptif

- Chaque tâche a une priorité **modifiable**
- Exécution de la tâche prête la plus prioritaire



Noury Bouraqadi – JM2L 2010



Systemes Multi-Robots

Collections & itérations

- Bibliothèque de collections très riche
- Itérer sur une collection = envoi de message





Langage *Puissant*

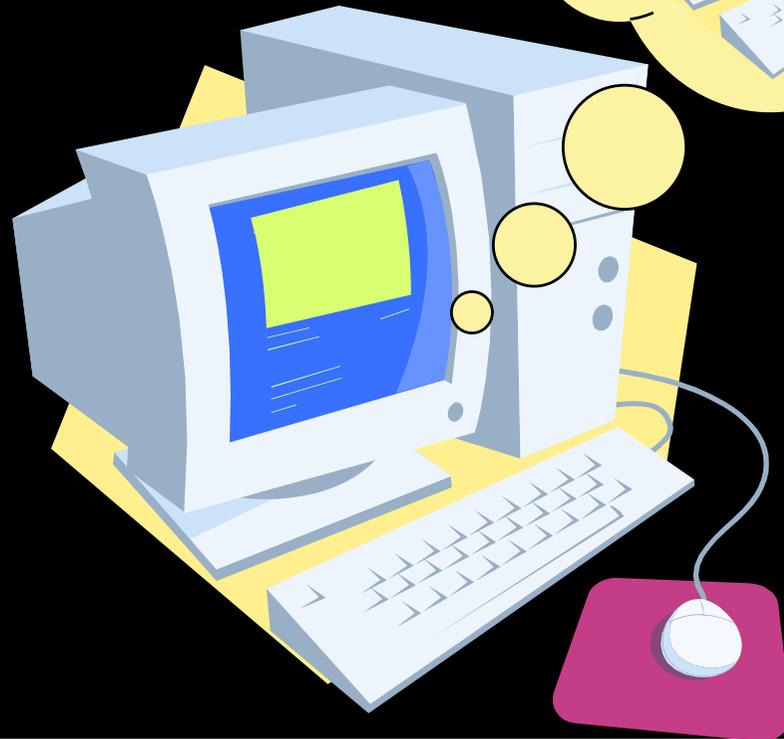
Réflexion

ΒΕΤΙΞΙΟΝ



Langage *Puissant*

Réflexion
ΚΕΤΙΧΙΟΥ



Noury Bouraqadi – JM2L 2010



Langage *Puissant*

Réflexion
REFLEXION

Classes

Objets représentant :

- *Entités du langage*
- *Environnement d'exécution*
- *Outils de développement*

Messages

Blocs

Méthodes



Langage *Puissant*

Réflexion
REFLEXION

Ordonnanceur

Objets représentant :

- *Entités du langage*
- *Environnement d'exécution*
- *Outils de développement*

*Pile
d'exécution*

Processus



Langage *Puissant*

Réflexion
REFLEXION

Noury Bouraqadi – JM2L 2010

Debugger

Objets représentant :

- Entités du langage
- Environnement d'exécution
- Outils de développement

Browser

Compilateur

Console



Langage ***Puissant***

Réflexion
REFLEXION

Toutes les classes
sont
modifiables



Langage *Puissant*

Réflexion
REFLEXION

Toutes les classes
sont
modifiables

Object

Metaclass

Class

Process

Message

Compiler

UndefinedObject

Debugger

Browser

BlockClosure

Boolean

Réflexion

REFLEXION

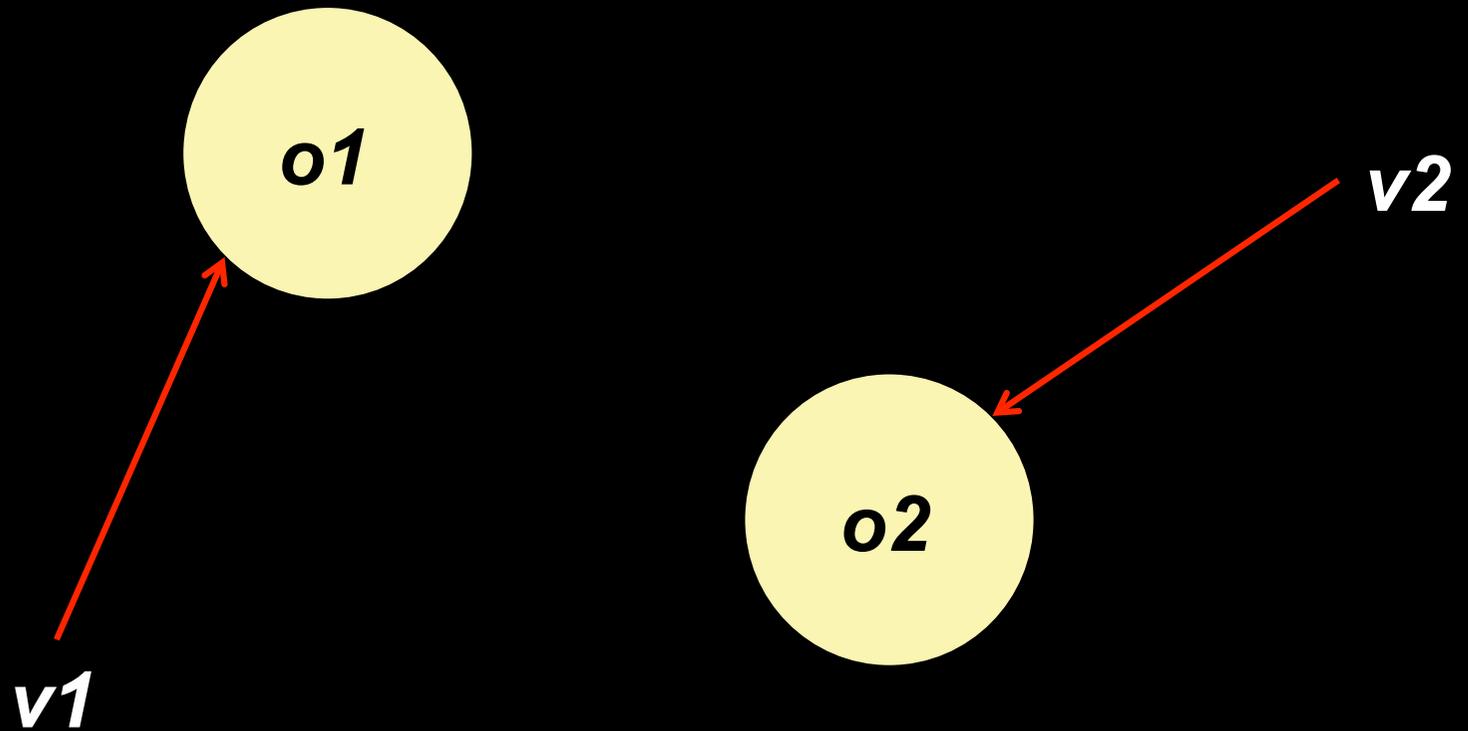
*Changer la
classe
d'un objet*

*Retrouver
les références
sur un objet*

*Echanger
l'identité de 2
objets*

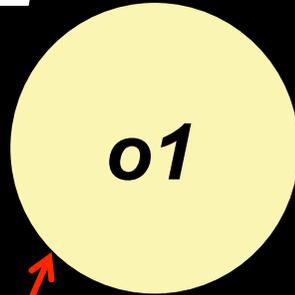
Méta-Opérations

Echange d'identité !



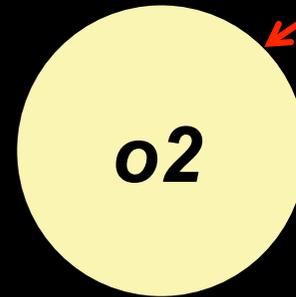
Echange d'identité !

become: o2



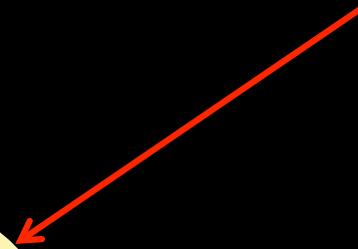
o1

v1

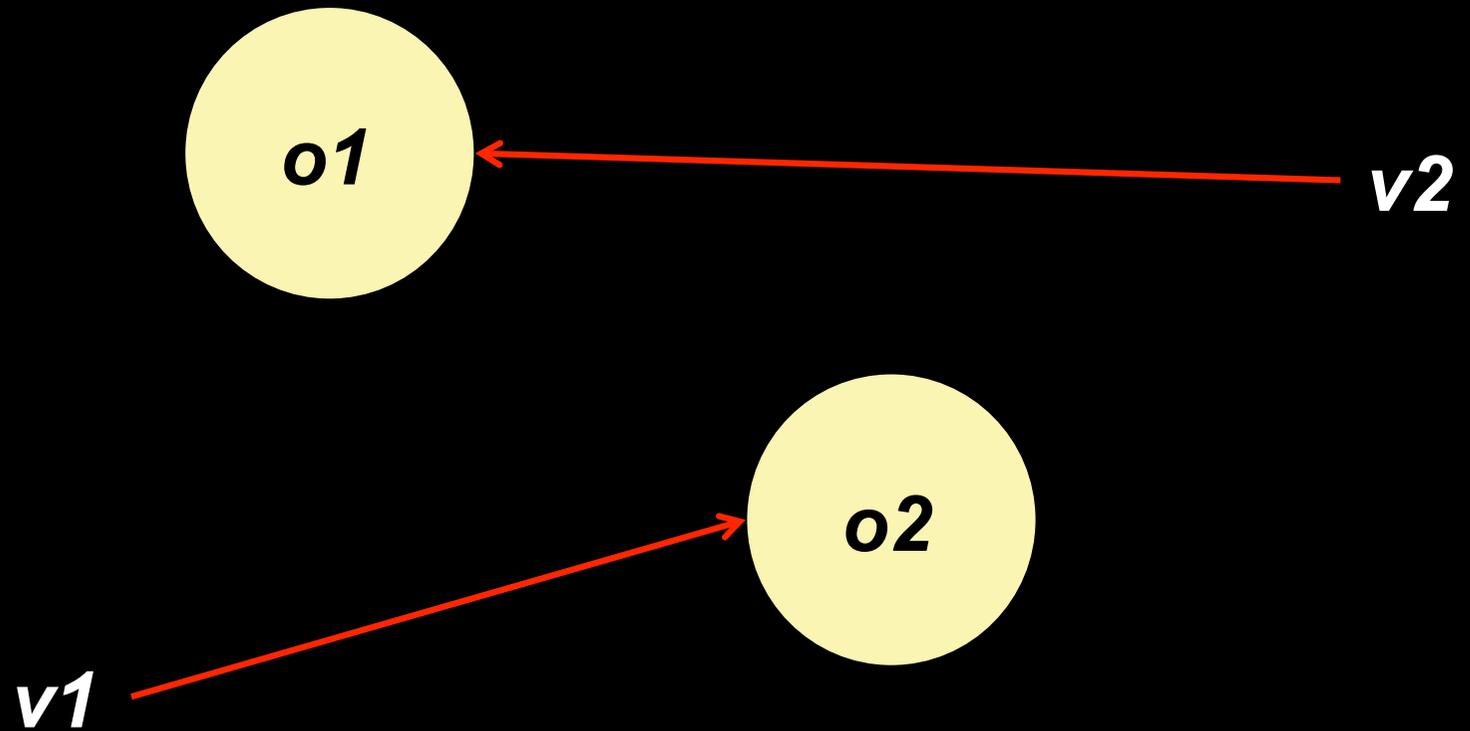


o2

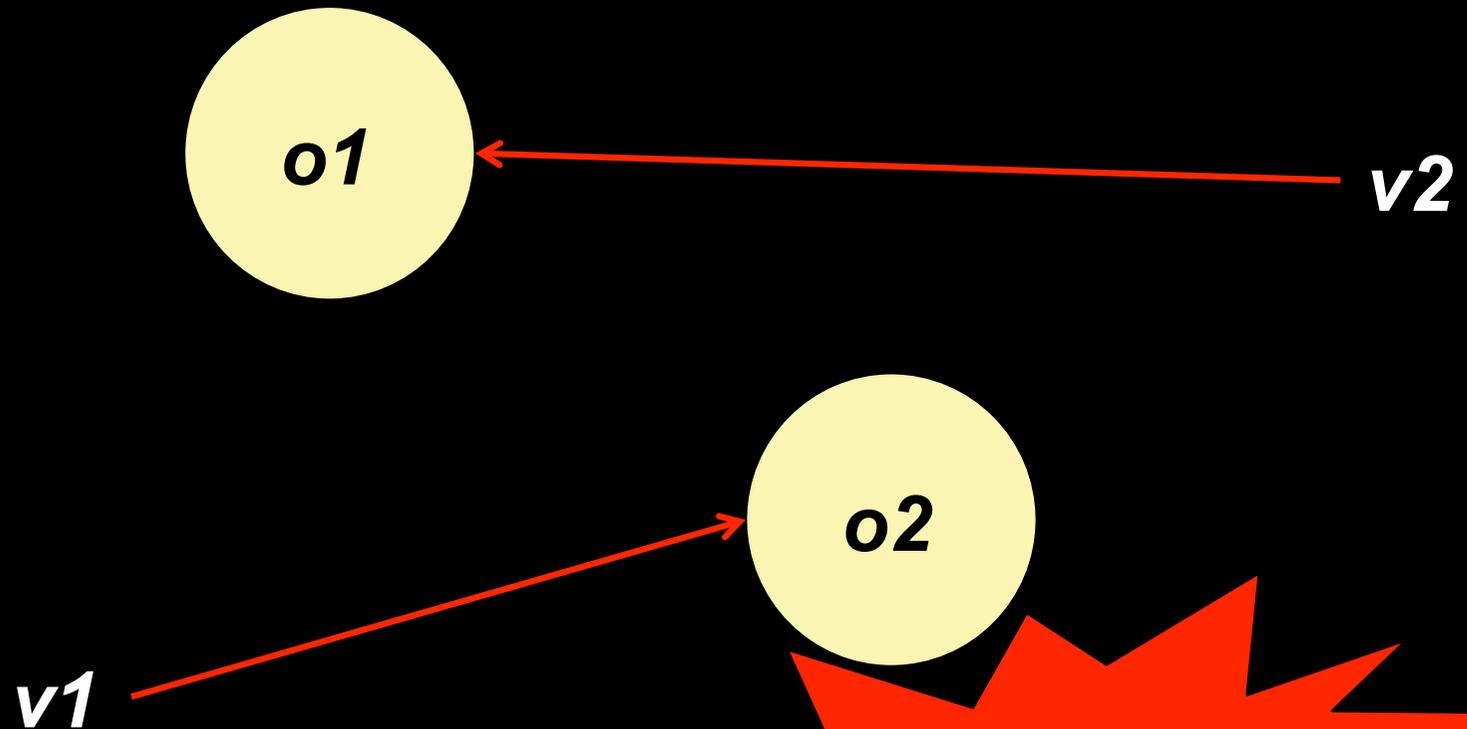
v2



Echange d'identité !



Echange d'identité !



Démo 9

Pour en savoir plus

Programmation Visuelle

- **Physical-eToys**



<http://tecnodacta.com.ar/gira/projects/physical-etoys/>

- **Phidgetlab**



<http://www.hpi.uni-potsdam.de/hirschfeld/projects/phidgetlab/>

- **SqueakBot**



<http://wiki.laptop.org/go/Projects/SqueakBot>

Smalltalks dédiés

- **NxTalk** : Lego Mindstorm

<http://www.hpi.uni-potsdam.de/hirschfeld/projects/nxtalk/>

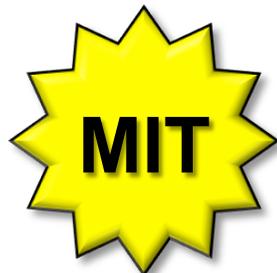


- **Smalltalk pour PIC**

<http://www.huv.com/uSeeker/smalltalk/pic.html>

- **FoxTalk**

<http://foxtalkbots.com/>





Robotique @ Douai

WifiBotST

<http://vst.mines-douai.fr/WifiBotST>

Articles, Code, Vidéos

<http://vst.mines-douai.fr/Robotics>

Pharo Open Source Smalltalk - Home

http://www.pharo-project.org/home

Download Pharo 1.1
For OS X, Linux, and Windows

Pharo

LE Site

Pharo is a clean, innovative, open-source Smalltalk-inspired environment

- pure object-oriented language
- stable core with large test suite
- good developer tools
- runs on all major platforms
- a clean look and feel
- low memory footprint

Version 1.1.1 (stable)

Pharo 1.1.1 was released on October 1, 2010. This maintenance release adds support for the new JIT VM named Cog. In addition it fixes *some important bugs* of Pharo 1.1. The versions of the external packages was not changed. For CogVM we recommend to use the latest version from www.mirandabanda.org/files/Cog/VM/

Version 1.2 alpha (unstable)

If you want to use all the new features and improvements that have been added to Pharo lately, and you don't mind living on the bleeding edge, [download](#) the current Pharo 1.2.

Getting started

- [Download Pharo](#)
- Run the executable after unzipping – no installation required!
- Follow the integrated, interactive tutorial
- [Join us and ask questions](#)
- [Get the Pharo By Example book \(free PDF available\)](#)
- [Watch the Screencasts](#)

Twitter news feed

Smalltalk 4 U 2: Getting Started with Pharo - <http://www.jarober.com/b...>
10:58 AM Oct 19th

[#Pharo Kernel. A 2.2MB...](#)

[New Chapter for #pharo book online. Omnibrowser <https://qforge.inria.fr/...>](#)
11:55 AM Aug 21st

Home
News
About
Success stories
Screenshots
Board

Download
Release 1.1
Release 1.0
Media files

Community
Issue Tracking
Contributors

License

Documentation
Tutorials/Books
Screencasts
FAQ

Pharo by Example

Search

Goals

- Smalltalk management system
- Package management
- Package self-test
- Package integration (or fork)

Preview

- A new compiler
- First-class packages
- Bootstrap from kernel image
- Rome

<http://www.pharo-project.org/>

LE Livre

Noury Bouraqadi – JM2L 2010



Andrew P. Black, Stéphane Ducasse, Oscar Nierstrasz, Damien Pollet
with Damien Cassou and Marcus Denker

Free
PDF



<http://pharobyexample.org/>

Start Pharo on GNU/Linux

Learn Smalltalk with ProfStef

Display Picasa photos

Programming with live objects

Monticello and Metacello introduction

How to contribute to Pharo, WorldMenu

The Lights Out game

Proportional Layout

SandstoneDb, simple ActiveRecord style

Les VIDEOS

<http://www.pharocasts.com/>

A vos agenda !

*Organisé
par
Linux Azur*

Atelier sur *Pharo* 

Vendredi 17 juin 2011

Atelier animé par *ramix*

*René
Mages*

Robots

Mobiles & Autonomes

avec

Pharo

Noury Bouraqadi – JM2L 2010

Noury Bouraqadi

<http://car.mines-douai.fr/noury>

JM2L – 27 nov 2010