FROM RESEARCH TO INDUSTRY

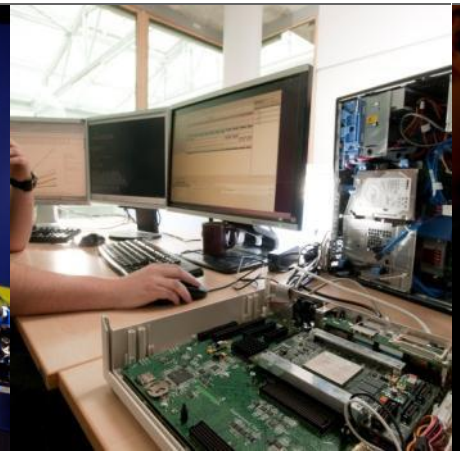
cea tech

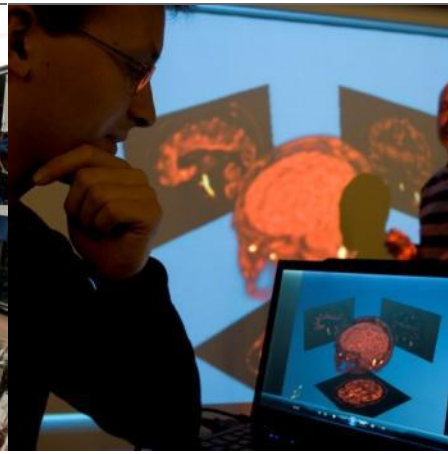
list

Projet P-RC2
Platform for Robot Controller Construction

Baptiste Gradoussoff & Benoît Milville

Laboratoire de Robotique Interactive





INSTITUT CARNOT CEA LIST

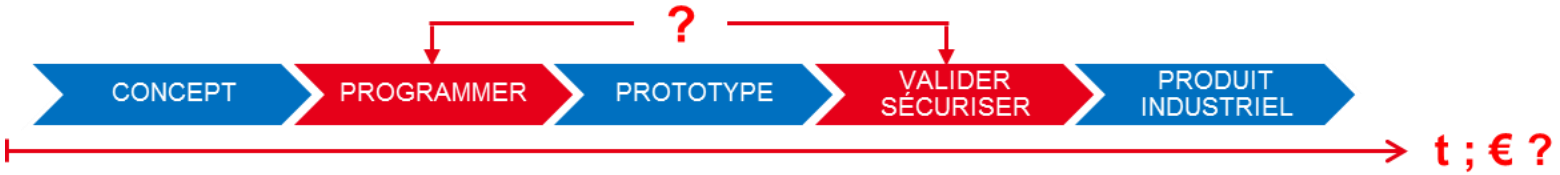
PLAN

1. Introduction
2. Positionnement du projet
3. Méthodologie
4. Architecture
5. Conclusion / Questions

INTRODUCTION
PROBLÉMATIQUE

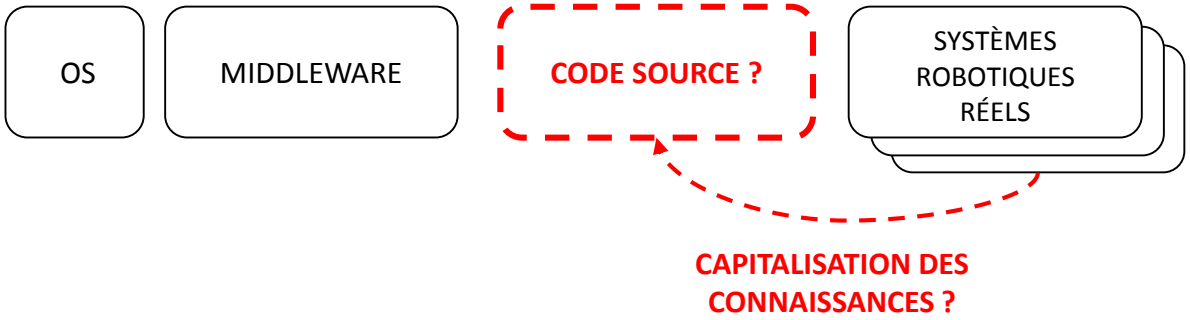
Problématique générale :

- Comment **faciliter le développement d'applications robotiques** académiques et industrielles à forte innovation ?
  - Comment **faciliter** la programmation des applications robotiques ?
  - Comment **valider et sécuriser** leur fonctionnement pour un environnement industriel ?
  - Comment **accélérer** les temps de développement, et **diminuer les coûts** associés ?



Verrou technologique :

- Capitalisation des connaissances, et notamment des briques logicielles réalisées à partir des middlewares robotiques



INTRODUCTION

LE PROJET P-RC2

Call PIA / LEOC (Programme Investissements d'Avenir / Logiciel Embarqué et Objets Connectés)

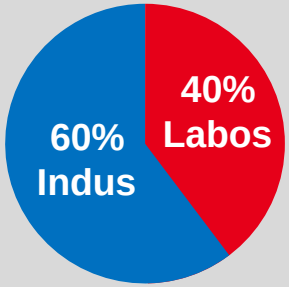
Consortium :

AKÉO+ ADVANCED MACHINE VISION — Leader

ARCURE be secure

BA SYSTEMES

SARRAZIN Technologies



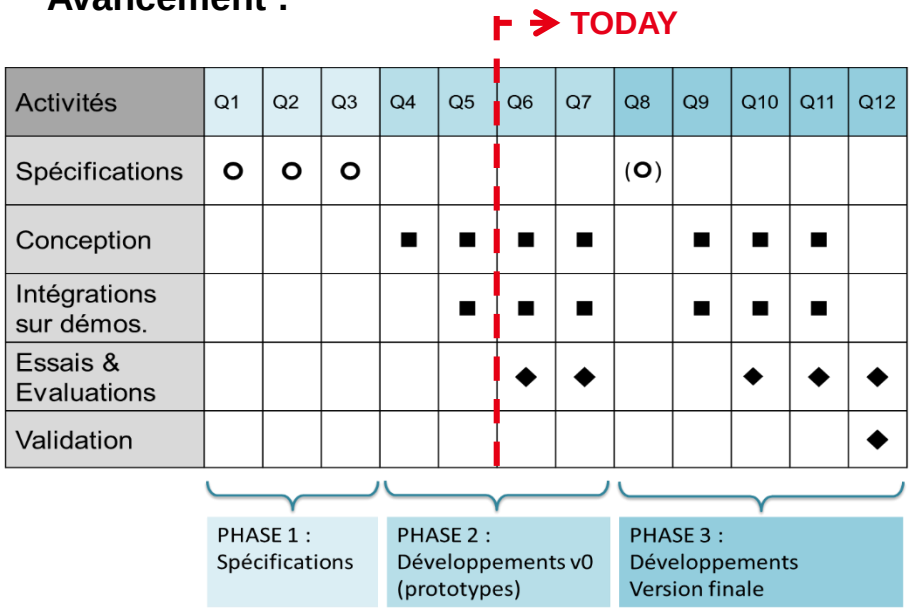
FROM RESEARCH TO INDUSTRY cea tech

LRI (Robotique)
LVIC (Vision)
LSL (Sûreté logicielle)

Jalons :

- Déc. 2014 : Lancement du projet
- Déc. 2016 : Premières démos
- Déc. 2017 : Fin projet
- 2019/2020 : Commercialisation

Avancement :



POSITIONNEMENT DU PROJET

3 piliers :

Généricité Modularité

- Toutes applications robotiques
- Architecture modulaire
- Réutilisation des développements précédents

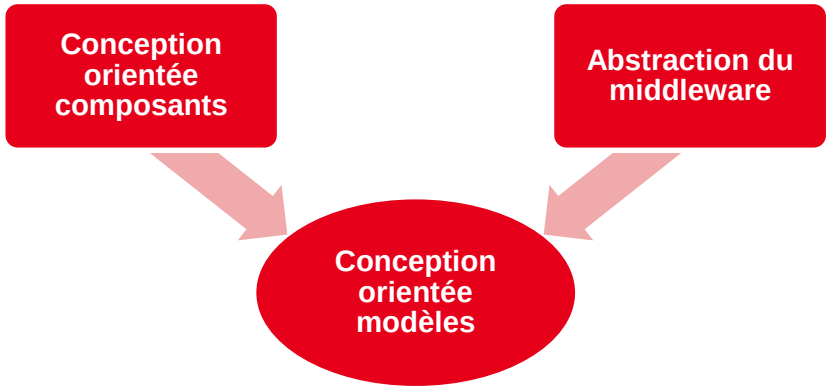
Ergonomie

- Packaging des outils, documentation
- Diffusion par un éditeur logiciel partenaire
- Communauté d'entraide

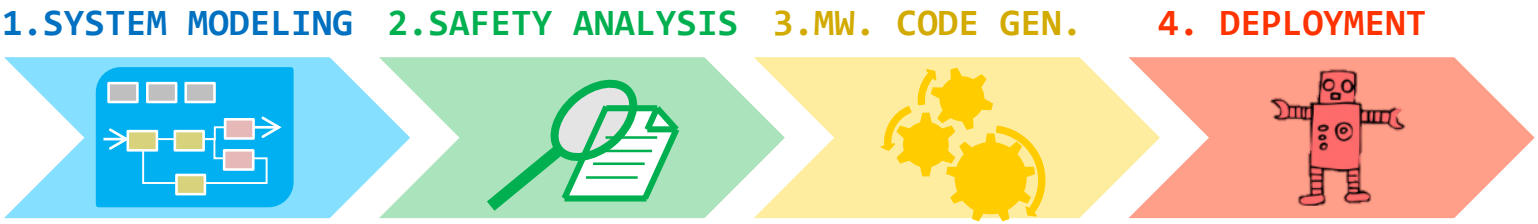
SdF

- Outils d'analyse de SdF (preuve formelle)
- Méthodologie associée

3 concepts clé :

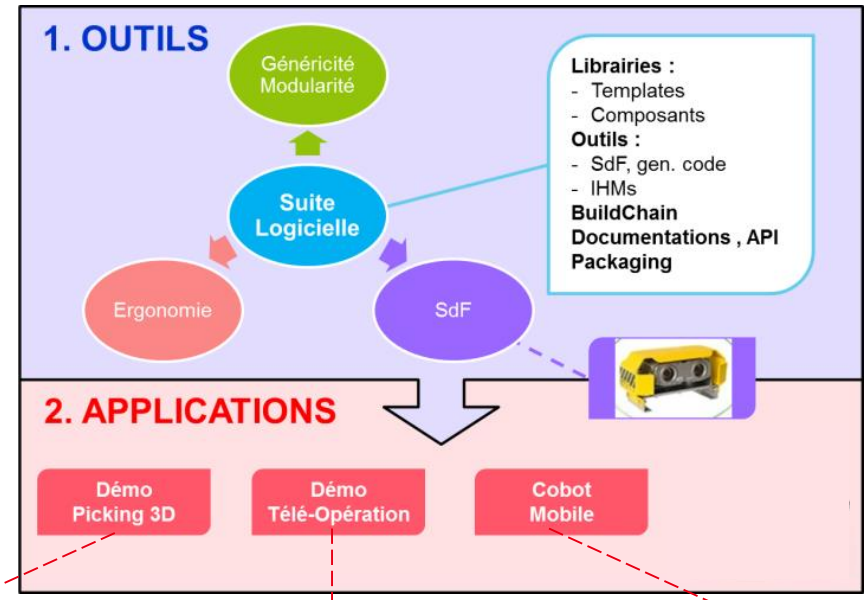


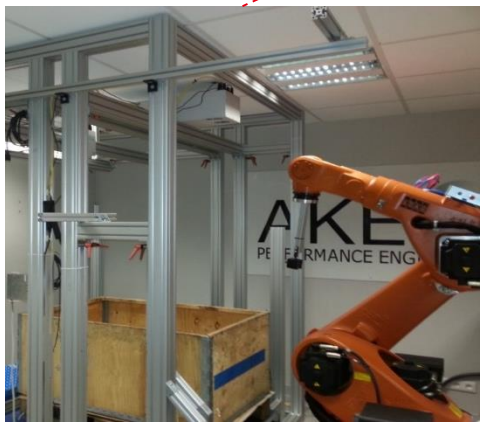
Workflow simplifié :

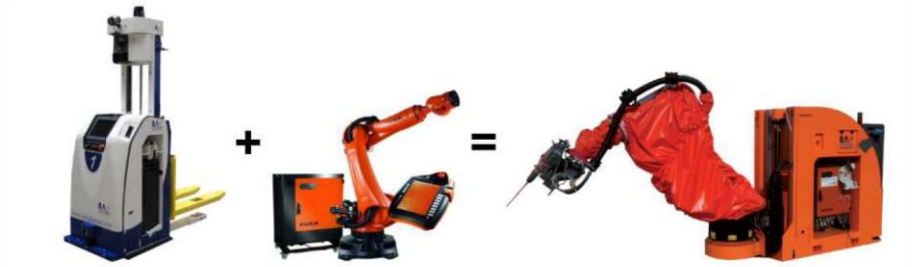


POSITIONNEMENT DU PROJET

Livrables :







P-RC2 : LA MÉTHODOLOGIE

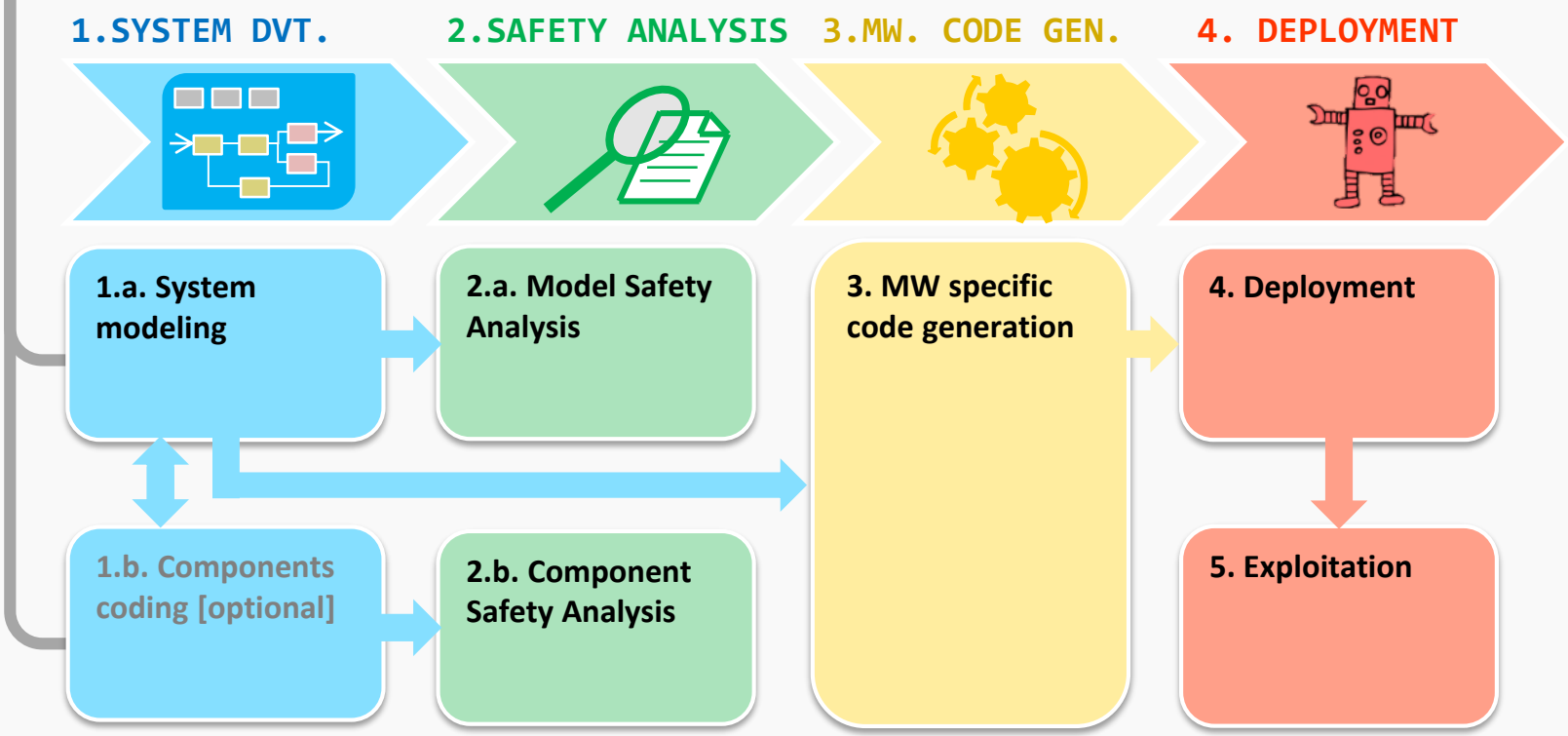
LE FLUX DE TRAVAIL

RESOURCES

- Models library
- Components library

- MW specific code generation templates

ACTIONS



TOOLS

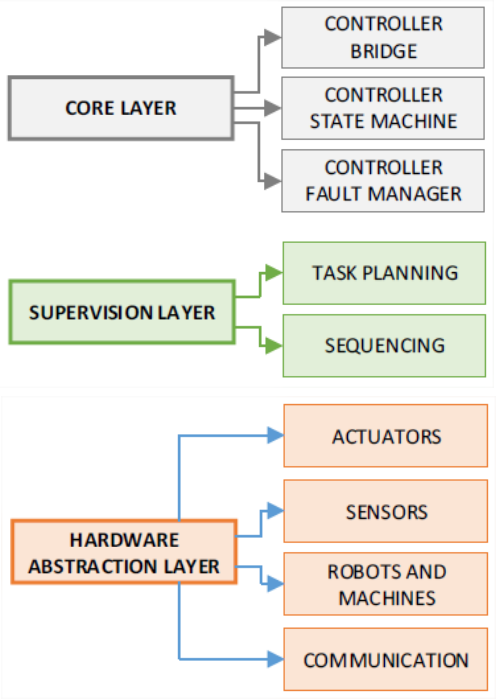
Papyrus* RobotML*	Frama-C Sophia*	Acceleo* or Xtend*	Robotics HMIs

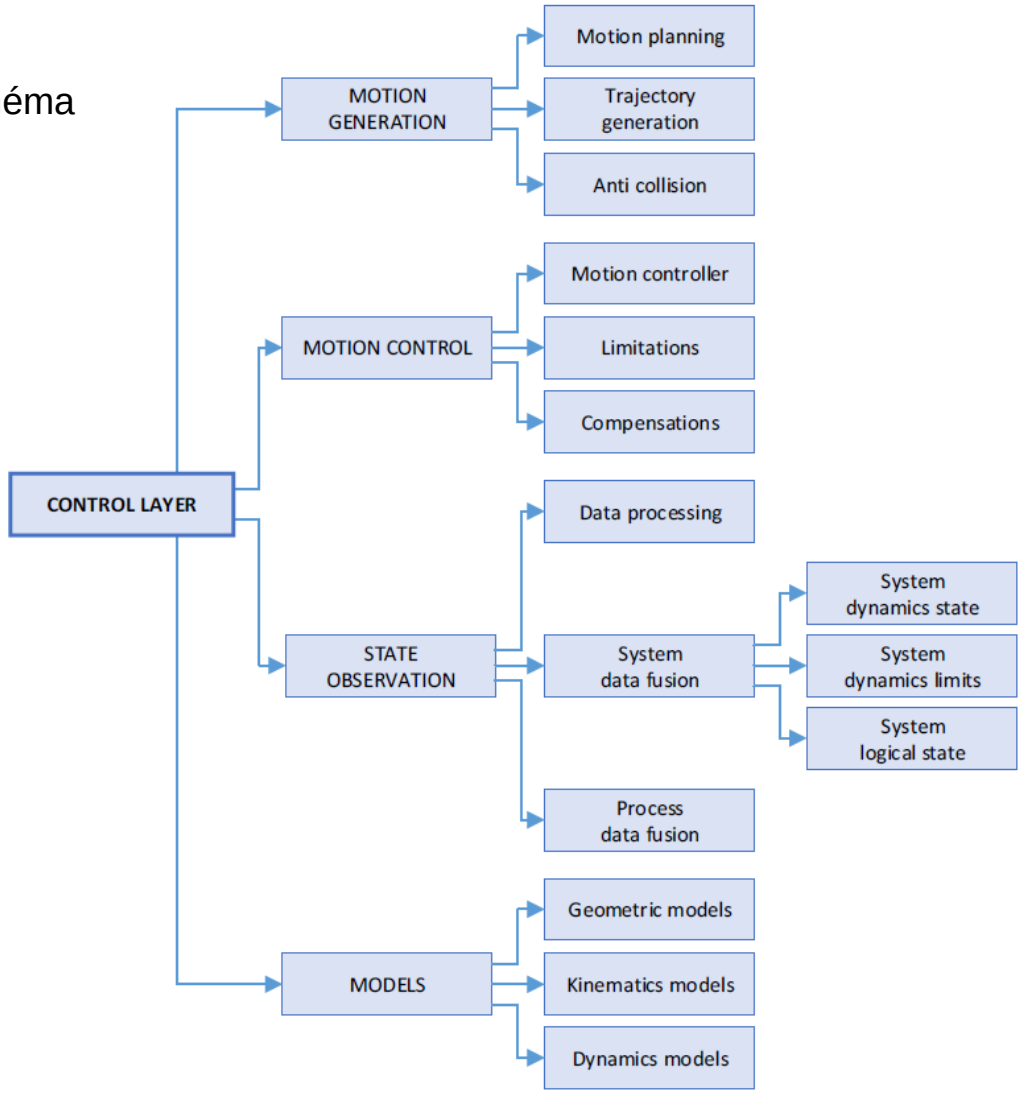
* Eclipse plugins

- **Classification des composants**

objectifs :

- granularité forte : ~15 composants par schéma
- focus sur la partie temps réel (t ≤ 50ms)





- **Standardisation des API composants**

Problématiques :

- **modularité :** composants d'origines diverses => besoin de standardiser les types métier pour garantir la compatibilité.
- **ergonomie :** besoin de standardiser les noms des API composants
- ergonomie : besoin de standardiser la partie générique de l'API contrôleur. (fonctions de type *powerOn(), getState(), getFaults(),* …)

Travaux en cours :

- **Spécification des API composants et contrôleurs**
  - Conventions de représentation (quaternions, …)
  - Types métiers de référence (pose, twist, wrench, …)
  - Types informatiques de référence
  - Conventions de nommage

- **Implémentation des spécifications**
  - Niveau modèle : implémentation des types métier dans le méta-modèle RobotML
  - Niveau code source : implémentation des types métiers (via librairies spécialisées : Eigen, LGSM, URDF…) et des *typekits* middlewares correspondants.

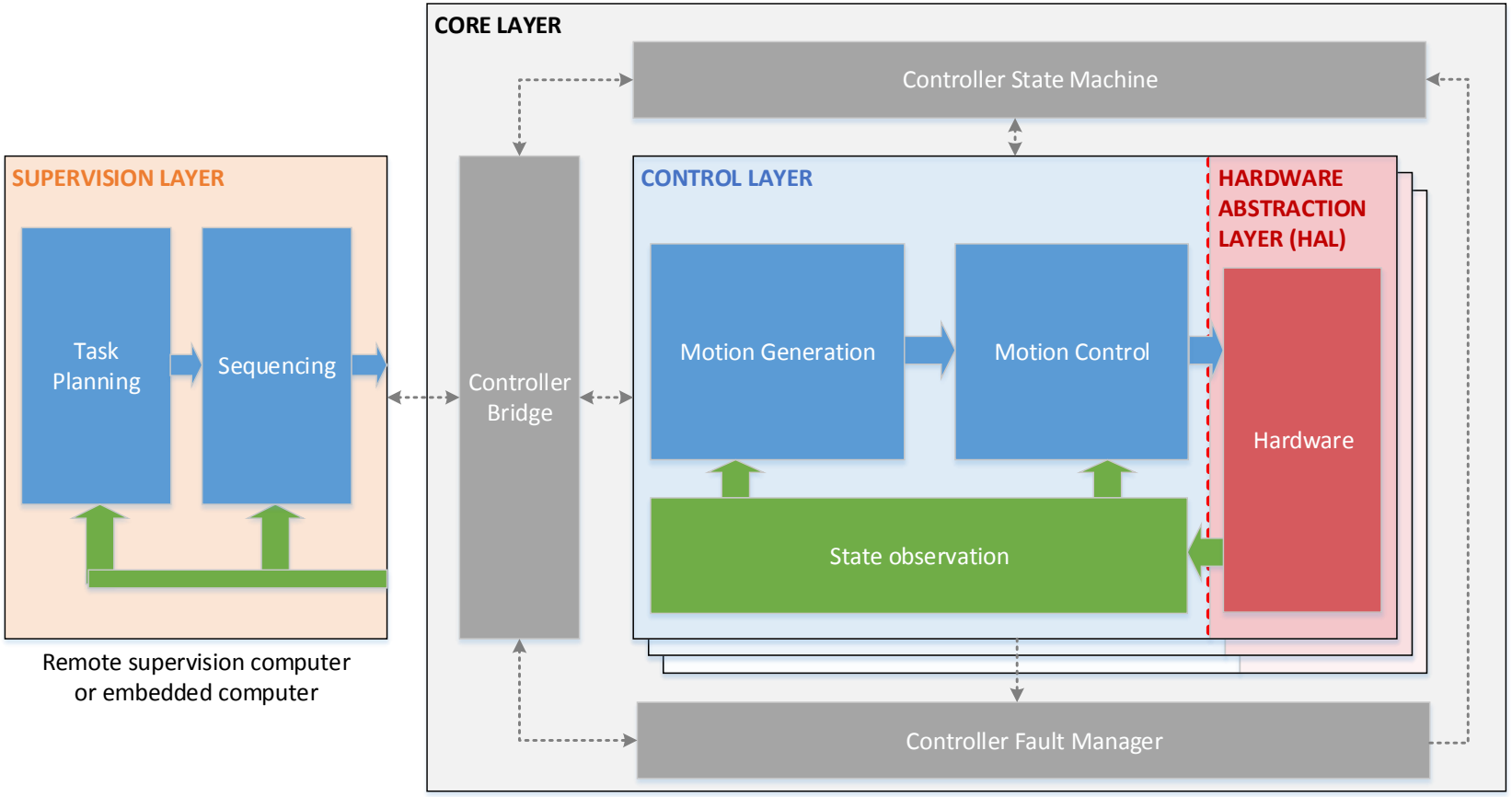
P-RC2 : L'ARCHITECTURE

ARCHITECTURE FONCTIONNELLE

1 système robotique = N modes de fonctionnement

1 Modèle de contrôleur =

- N Schémas d'architecture
- 1 HFSM (Hierarchical Finite State Machine) de supervision des modes de fonctionnement

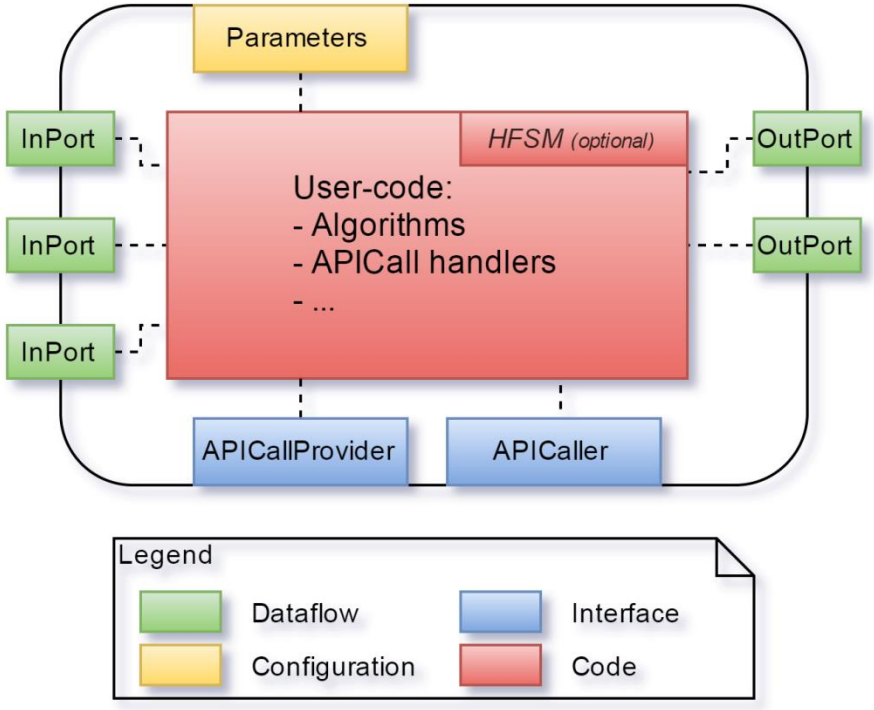


P-RC2 : L'ARCHITECTURE

ABSTRACTION DU MIDDLEWARE

Concept de composant P-RC2:
- Ports
- Paramètres
- Méthodes
- (Machine à états)

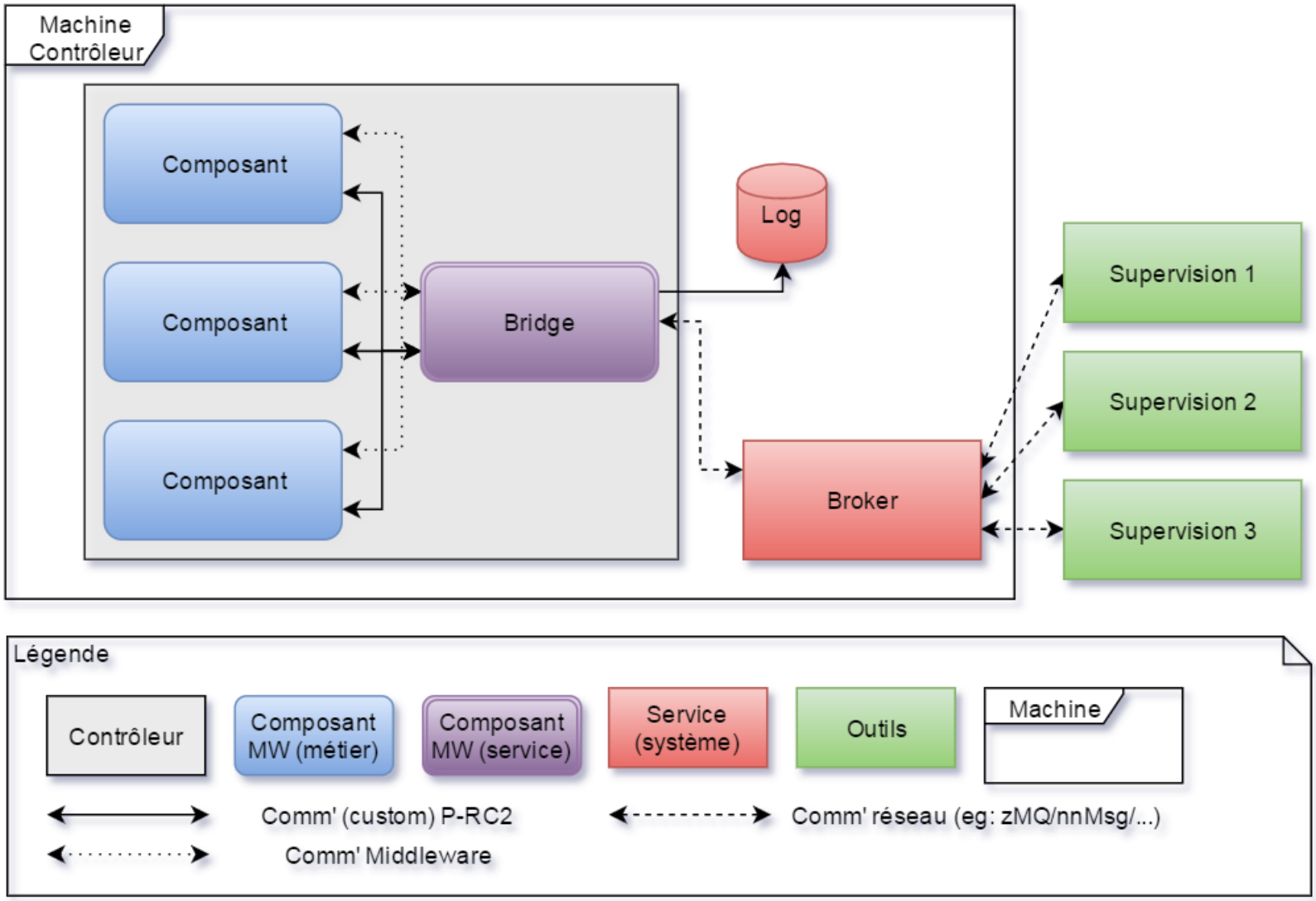
A « traduire » pour les *middlewares*. Par exemple:



P-RC2	OROCOS 2.8	ROS (Jade)
Component	TaskContext	Node
InPort	InputPort	Subscriber
OutPort	OutputPort	Publisher
Parameter	Property	Parameter
APICallProvider	Operation	Service / Action
APICaller	OperationCaller	ServiceClient / ActionClient
APICallReq	SendHandle	GoalHandle

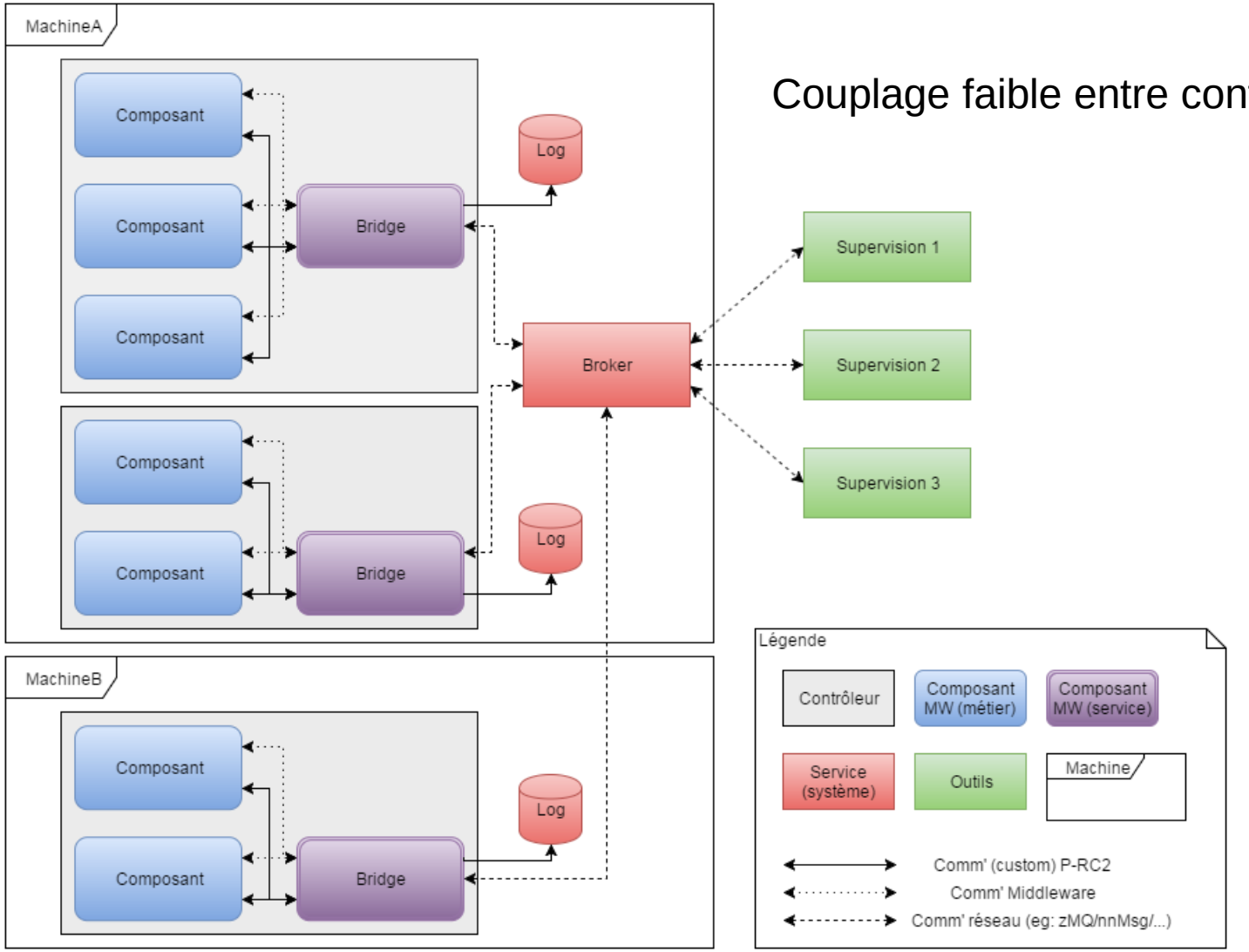
P-RC2 : L'ARCHITECTURE

ARCHITECTURE D'UN CONTRÔLEUR



P-RC2 : L'ARCHITECTURE

ARCHITECTURE AVEC PLUSIEURS CONTRÔLEURS



Couplage faible entre contrôleurs

CONCLUSION

- **Travaux futurs**
  - Modélisation
    - Implémentation de la standardisation dans les outils
    - Création de contenu
  - Logiciel
    - Services contrôleur et système
    - Interfaces de supervision
  - Caractérisation
    - Performances
    - Ergonomie
  - Documentation
  - Démonstrateurs

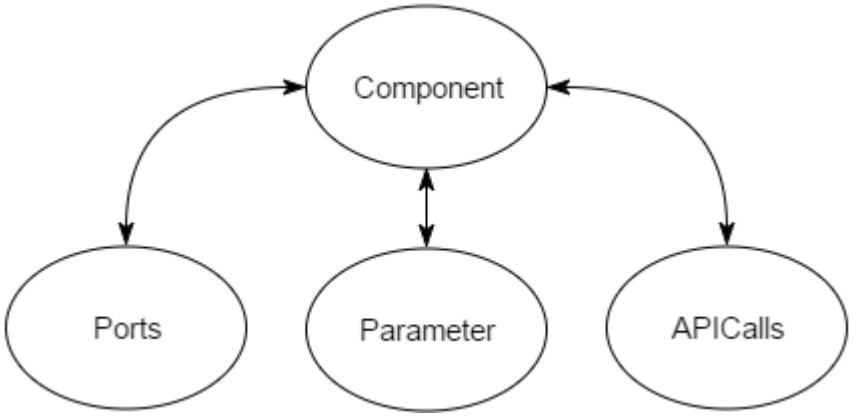
- **Problématiques ouvertes / incertitudes**
  - Robustesse de la standardisation des composants
  - gestion des transitions entre modes de fonctionnement : capacité à implémenter les transitions, capacité à réaliser des transitions à chaud (actionneurs commandés), sûreté de fonctionnement liée au redéploiement

ANNEXES

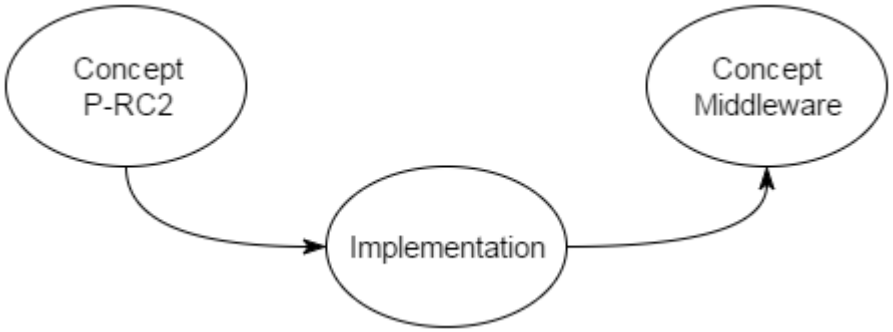
P-RC2 : L'ARCHITECTURE

ABSTRACTION DU MIDDLEWARE

- **Hiérarchie**

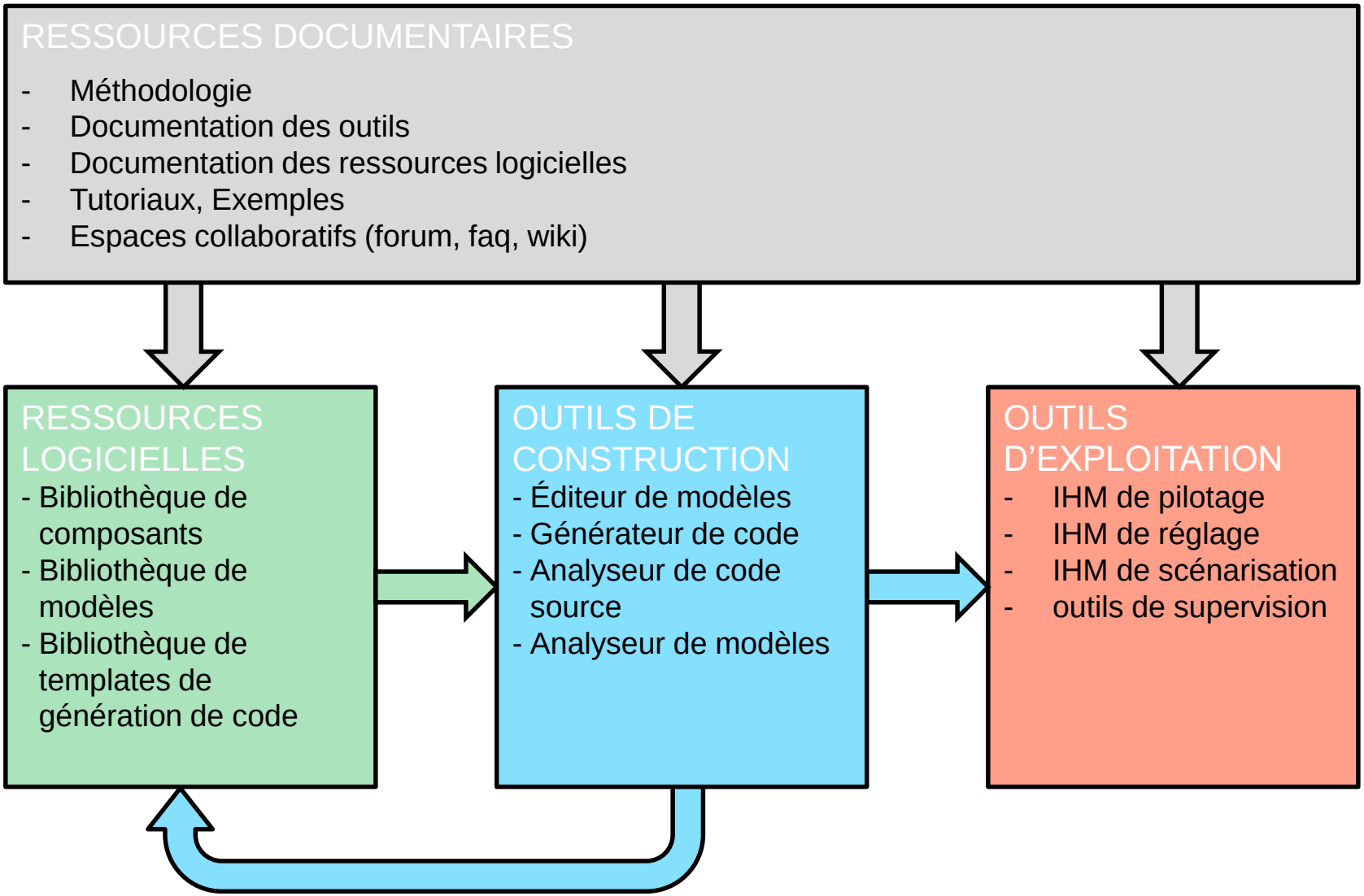


- **Concepts**



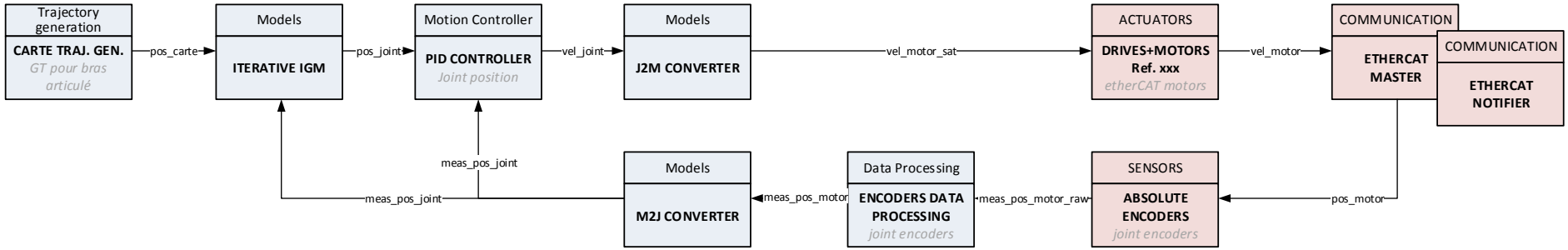
- A la compilation du concept P-RC2, l'implémentation n'est pas connue !
- Couplage relaché au prix d'une indirection pour chaque appel de méthode.

P-RC2 : LA BOITE À OUTILS



P-RC2: MODELING CONTENTS

- **Controller model**
  - 1 Controller Model =
    - N Architecture Diagrams (1 for each working mode)
    - 1 general HFSM (Hierarchical Finite State Machine) describing the controller behaviour between the N working modes.
- **Architecture diagrams**



  - Components interface :
    - Ports (I/O) : name, type
    - APICalls, provided or required. (~ I/O Service ports) : prototype (name, arguments name & type, return type)
    - parameters : name & type
  - Functional architecture: instanciation, ports connections, APICalls connections
  - Deployment configuration : components scheduling (activity, period), threads/processes distribution
- **Controller HFSM**
  - States : n..1 correspondance with the N architecture diagrams (/!\ n ≠ N) .
  - Transitions : events, guards, effects