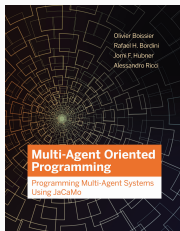


# Multi-Agent Oriented Programming

for Engineering Distributed Intelligent Systems



---

Olivier Boissier<sup>1</sup>, Jomi F. Hübner<sup>2</sup>

(1) Mines Saint-Etienne/Institut Mines Télécom/LIMOS UMR 6158, France

(2) Federal University of Santa Catarina, Brazil

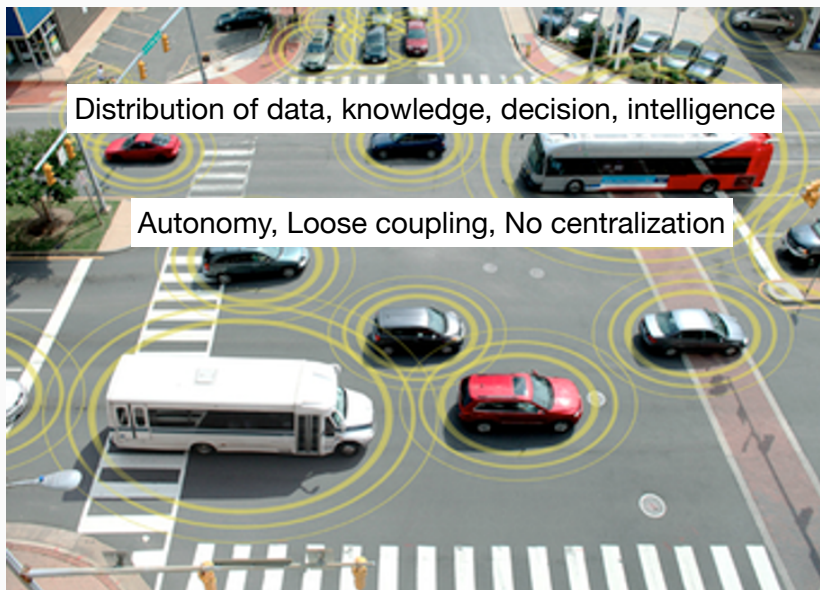
SAP Inspiration Session — June 23, 2022

# Distributed Intelligent Systems: Requirements

Distribution of data, knowledge, decision, intelligence



# Distributed Intelligent Systems: Requirements



Distribution of data, knowledge, decision, intelligence

Autonomy, Loose coupling, No centralization

# Distributed Intelligent Systems: Requirements

An aerial photograph of a city intersection. Several vehicles, including cars and buses, are visible. Each vehicle is surrounded by concentric yellow circles, representing sensor waves or communication ranges. The circles overlap, illustrating a distributed network of intelligent agents.

Distribution of data, knowledge, decision, intelligence

Autonomy, Loose coupling, No centralization

Openness, Long liveness, Heterogeneity

# Distributed Intelligent Systems: Requirements

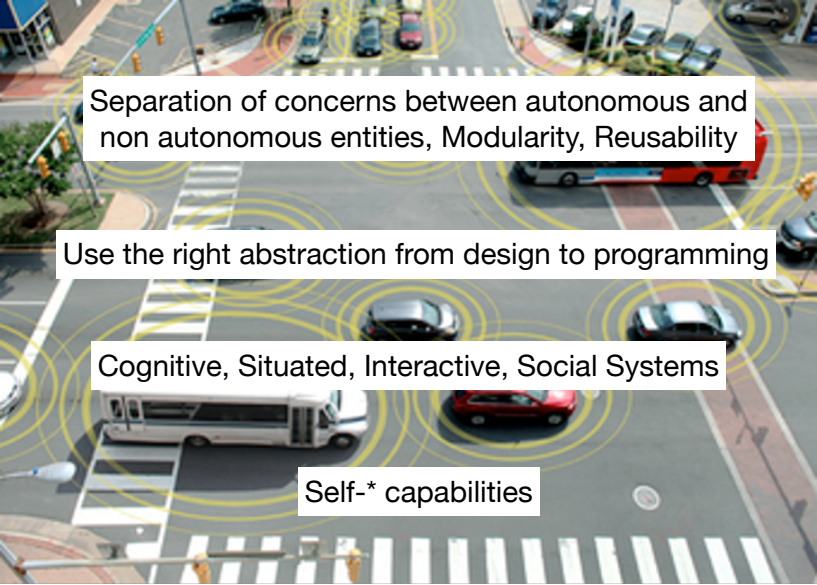
An aerial photograph of a city intersection. Several vehicles, including cars and buses, are visible. Each vehicle is surrounded by concentric yellow circles, representing sensor waves or communication ranges. The text boxes are overlaid on the image.

Distribution of data, knowledge, decision, intelligence

Autonomy, Loose coupling, No centralization

Openness, Long liveness, Heterogeneity

Adaptation, Resilience, Agility



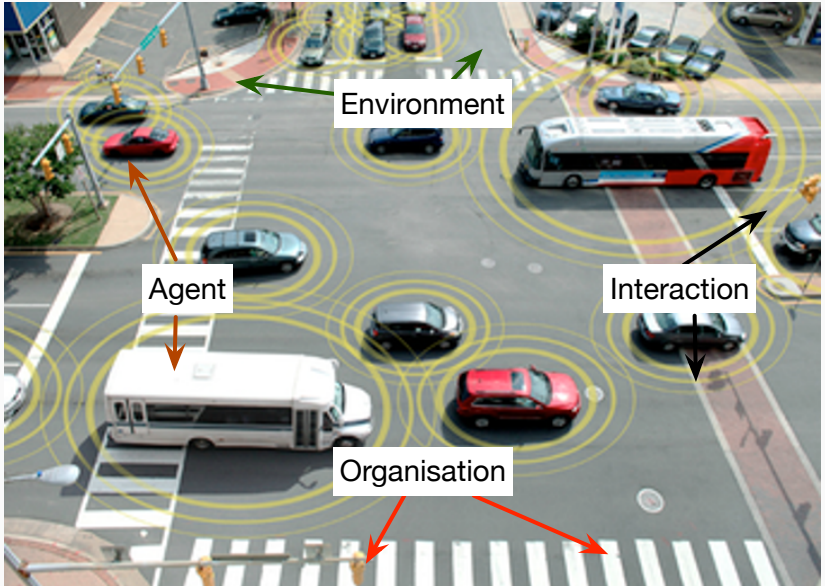
Separation of concerns between autonomous and non autonomous entities, Modularity, Reusability

Use the right abstraction from design to programming

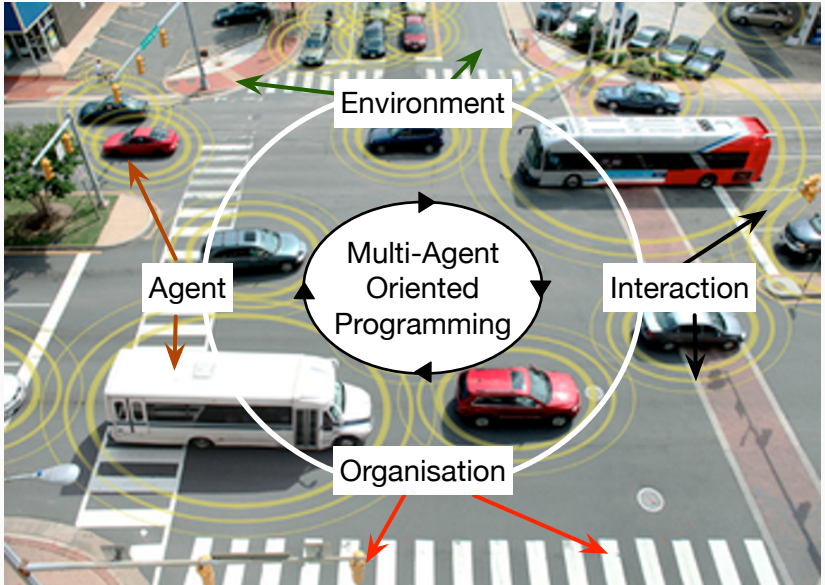
Cognitive, Situated, Interactive, Social Systems

Self-\* capabilities

# Outline

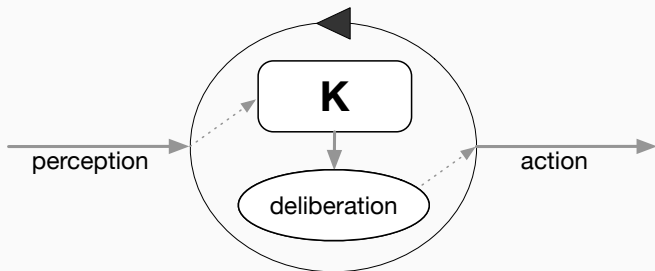


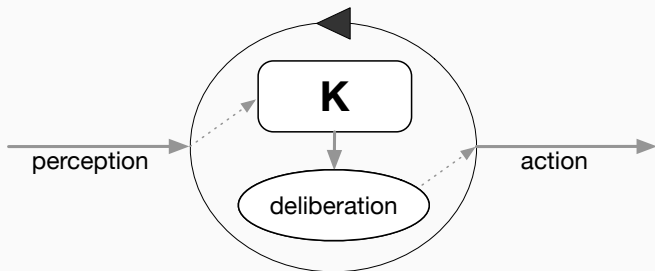
# Outline











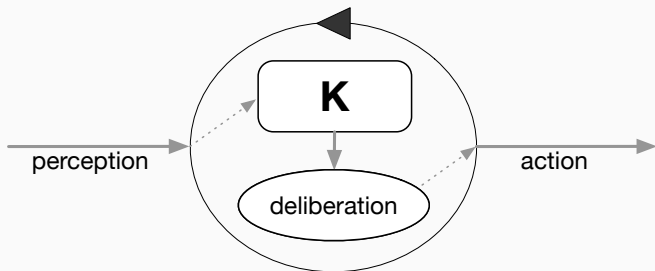
[reasoning cycle]

**while true do**

$K \leftarrow K \pm perception()$

$P \leftarrow deliberation(K)$

$act(P)$



to **program** an agent is to define **K**

deliberation  $\rightsquigarrow$  **autonomy**

**Beliefs** : information about the environment, other agents, itself, application, ....

`temperature(20) .`

`happy(bob) .`

**Goals** : the agent objectives

`!temperature(20) .`

`!happy(bob) .`

**Plans** :

**Beliefs** : information about the environment, other agents, itself, application, ....

`temperature(20) .`

`happy(bob) .`

**Goals** : the agent objectives

`!temperature(20) .`

`!happy(bob) .`

**Plans** :

**Beliefs** : information about the environment, other agents, itself, application, ....

```
temperature(20) .
```

```
happy(bob) .
```

**Goals** : the agent objectives

```
!temperature(20) .
```

```
!happy(bob) .
```

**Plans** : specifies how goals can be achieved

```
+!temperature(20) <- turn_on(ac) .
```

```
+!happy(bob) <- kiss(bob) .
```

**Beliefs** : information about the environment, other agents, itself, application, ....

```
temperature(20).
```

```
happy(bob).
```

**Goals** : the agent objectives

```
!temperature(20).
```

```
!happy(bob).
```

**Plans** : specifies how goals can be **achieved**

```
+!temperature(20) <- turn_on(ac).
```

```
+!happy(bob) <- kiss(bob).
```

specifies **reactions** to mental state changes

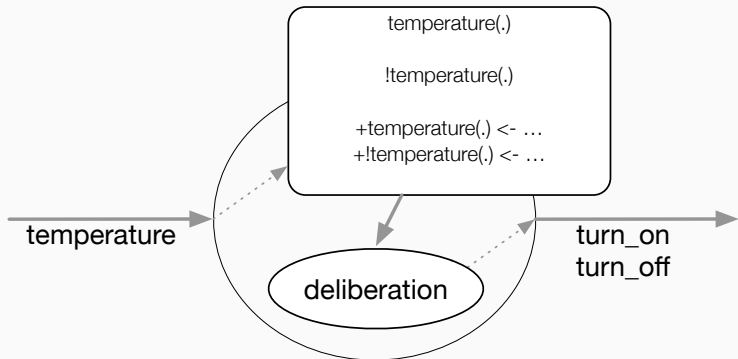
```
+temperature(10) <- !temperature(20).
```

```
-happy(bob) <- !happy(bob).
```



Beliefs, goals, and plans are provided by

- perception: in the case of beliefs
- developers: initial mental state of the agent
- other agents: by communication
- the agent itself: by reasoning or learning



## Agent Programming (in JaCaMo) ii

```
+temperature(30) <- !temperature(20).  
+!temperature(20) <- turn_on(ac).
```

```
+temperature(30) <- !temperature(20).
```

```
+temperature(20) <- turn_off(ac).
```

```
+!temperature(20) <- turn_on(ac).
```

```
// initial belief, given by the developer
preferred_temp(20).

// reaction to changes in the temperature
+temperature(T) : preferred_temp(P) & math.abs(P-T) > 2
  <- !temperature(P).
+temperature(T) : preferred_temp(T)
  <- turn_off(ac).

// plans to achieve some temperature
+!temperature(P) : temperature(T) & T > P
  <- turn_on(ac).
```

```
// initial belief, given by the developer
preferred_temp(20).

// initial goal, given by the developer
!keep_temperature.

// maintenance the goal pattern
+!keep_temperature
    : temperature(T) & preferred_temp(P) & T > P
    <- turn_on(ac);
    !keep_temperature.
+!keep_temperature
    : temperature(T) & preferred_temp(P) & T <= P
    <- turn_off(ac);
    !keep_temperature.
```

- **reactivity**: even when achieving some goals
- **pro-activity**: new goals can be created
- **long-term goals**: agents are committed to achieve goals
- **context awareness**: plans are selected based on the circumstances
- **transparency**: we can trace back the reasons for an action
- **sound theoretical background** for agent architectures:
  - practical reasoning [Bratman, 1987]
  - intentions [Cohen and Levesque, 1987]
  - BDI [Rao and Georgeff, 1995]
  - ...

- **reactivity**: even when achieving some goals
- **pro-activity**: new goals can be created
- **long-term goals**: agents are committed to achieve goals
- **context awareness**: plans are selected based on the circumstances
- **transparency**: we can trace back the reasons for an action
- **sound theoretical background** for agent architectures:
  - practical reasoning [Bratman, 1987]
  - intentions [Cohen and Levesque, 1987]
  - BDI [Rao and Georgeff, 1995]
  - ...



- **reactivity**: even when achieving some goals
- **pro-activity**: new goals can be created
- **long-term goals**: agents are committed to achieve goals
- **context awareness**: plans are selected based on the circumstances
- **transparency**: we can trace back the reasons for an action
- **sound theoretical background** for agent architectures:
  - practical reasoning [Bratman, 1987]
  - intentions [Cohen and Levesque, 1987]
  - BDI [Rao and Georgeff, 1995]
  - ...

- **reactivity**: even when achieving some goals
- **pro-activity**: new goals can be created
- **long-term goals**: agents are committed to achieve goals
- **context awareness**: plans are selected based on the circumstances
- **transparency**: we can trace back the reasons for an action
- **sound theoretical background** for agent architectures:
  - practical reasoning [Bratman, 1987]
  - intentions [Cohen and Levesque, 1987]
  - BDI [Rao and Georgeff, 1995]
  - ...

- **reactivity**: even when achieving some goals
- **pro-activity**: new goals can be created
- **long-term goals**: agents are committed to achieve goals
- **context awareness**: plans are selected based on the circumstances
- **transparency**: we can trace back the reasons for an action
- sound **theoretical background** for agent architectures:
  - practical reasoning [Bratman, 1987]
  - intentions [Cohen and Levesque, 1987]
  - BDI [Rao and Georgeff, 1995]
  - ...

- **reactivity**: even when achieving some goals
- **pro-activity**: new goals can be created
- **long-term goals**: agents are committed to achieve goals
- **context awareness**: plans are selected based on the circumstances
- **transparency**: we can trace back the reasons for an action
- sound **theoretical background** for agent architectures:
  - practical reasoning [Bratman, 1987]
  - intentions [Cohen and Levesque, 1987]
  - BDI [Rao and Georgeff, 1995]
  - ...

## A note about “Control”

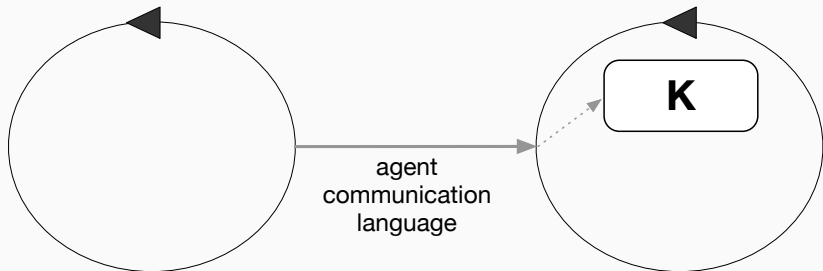
Agents **encapsulates** and control their own (and influence the others)

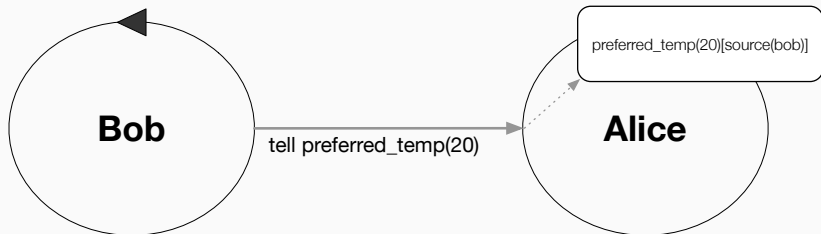
- beliefs
- goals
- plans

By doing so they control their behaviour  $\rightsquigarrow$  **autonomy**

The developer provides initial values of these elements and thus also influence the behaviour of the agent

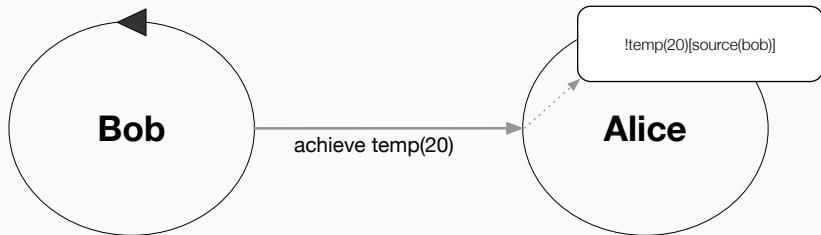
# Agent-Agent Communication





A message has:

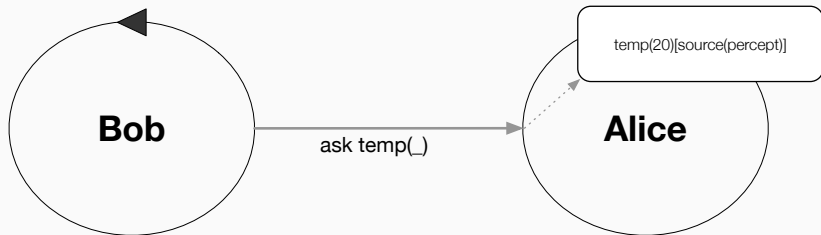
- an intention (tell, ask, achieve, ...)
- a content (belief, goal, plan)



A message has:

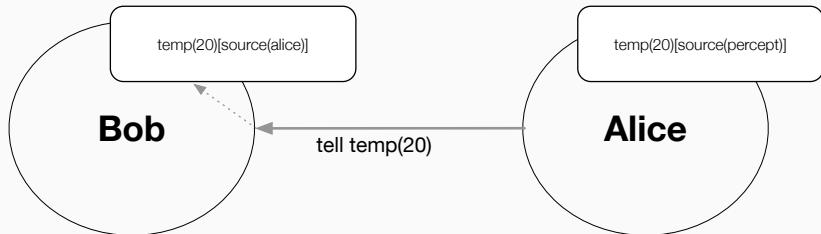
- an intention (tell, ask, achieve, ...)
- a content (belief, goal, plan)





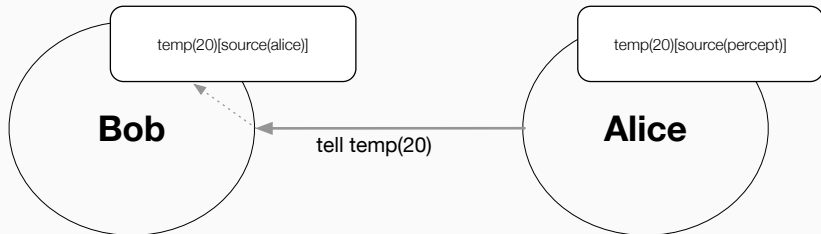
A message has:

- an intention (tell, ask, achieve, ...)
- a content (belief, goal, plan)



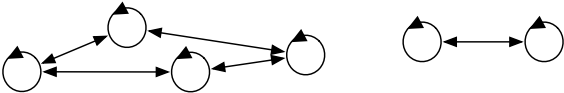
A message has:

- an intention (tell, ask, achieve, ...)
- a content (belief, goal, plan)

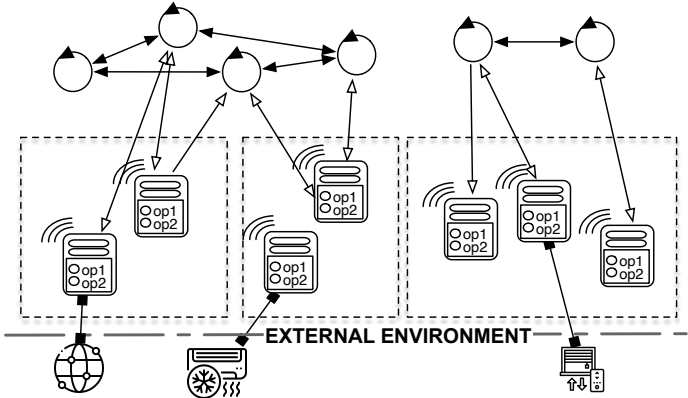


- we are not programming computers,  
we are programming agents, which are based on knowledge
- communication is not about data exchange, but  
knowledge sharing

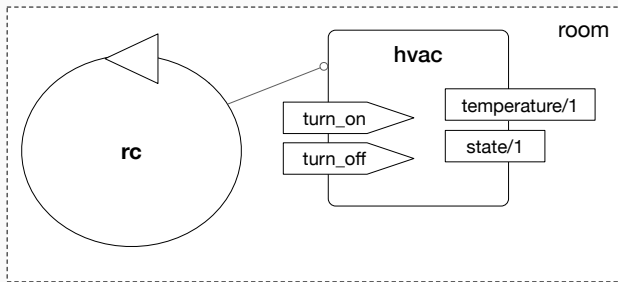
# Environment



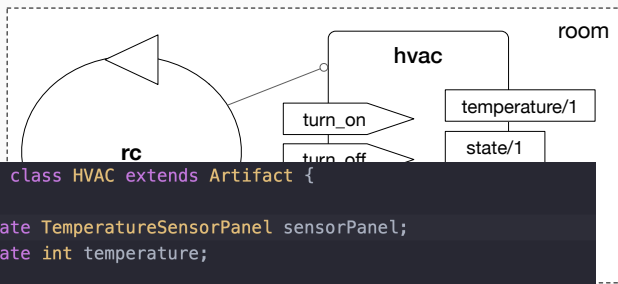
# Environment



# Environment Programming (in JaCaMo)



# Environment Programming (in JaCaMo)



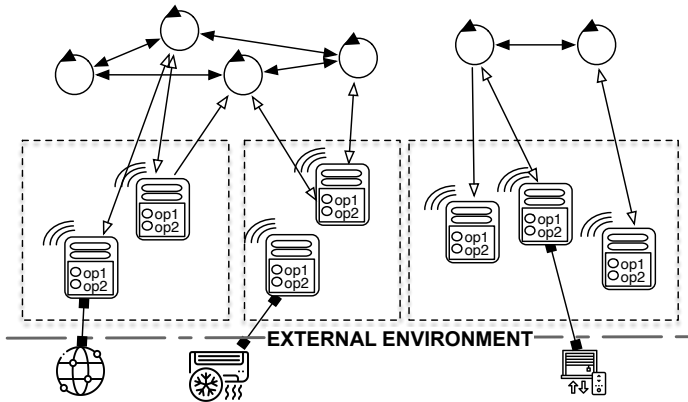
```
5 public class HVAC extends Artifact {
6
7 | private TemperatureSensorPanel sensorPanel;
8 private int temperature;
9
10 > void init(int temp){=
18
19 > @OPERATION void turn_on(Object device) {=
26
27 > @OPERATION void turn_off(Object device) {=
34
35 > @INTERNAL_OPERATION void updateTemperatureProc(int step){=
```

# Features of MAS Environment

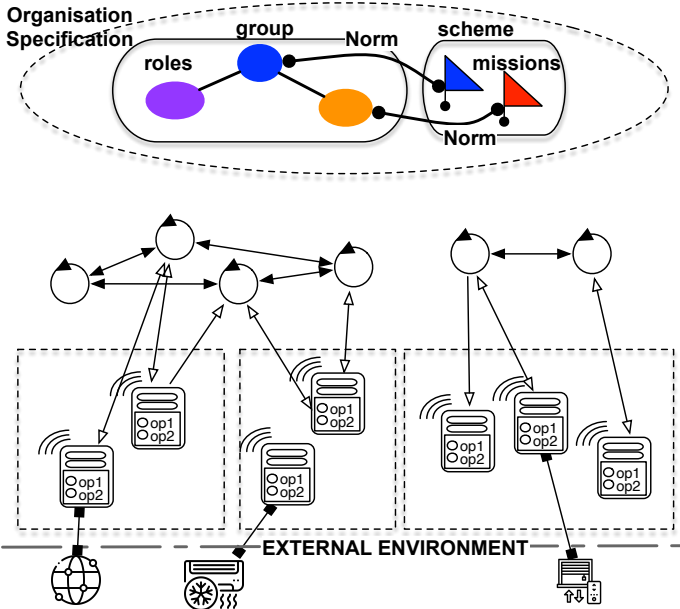
- **shared** among agents: indirect interaction
- **situatedness**: agents can move to different places and so change available actions and perception
- agents can **build** their (virtual) environment
- **separation of concerns**: autonomous entities vs tools
  
- artifacts are not objects: atomic and transactional operations, conceived for agents



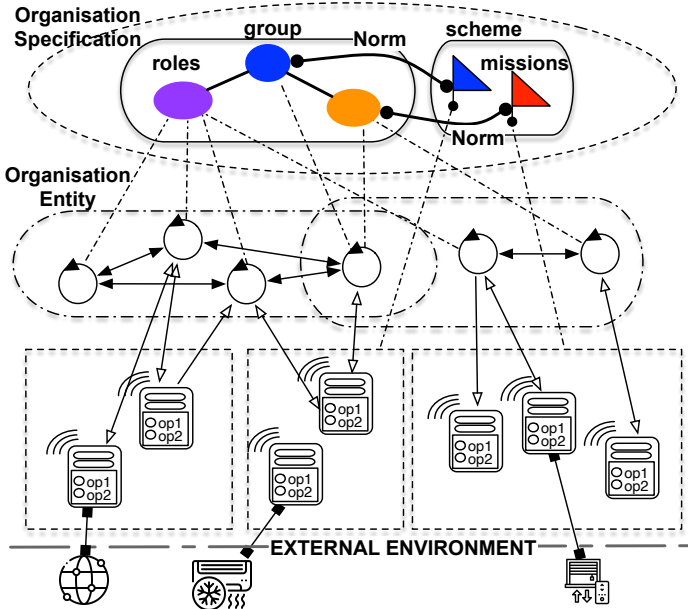
# Organisation



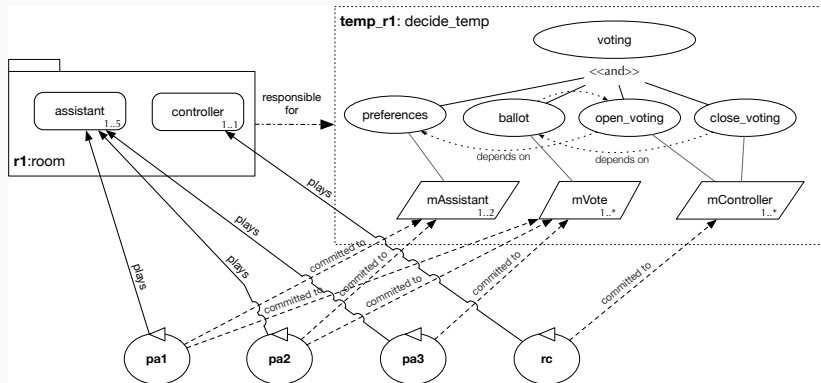
# Organisation



# Organisation



# Declarative Organisation Programming



- Structural patterns (groups (`r1:room`), roles (`assistant`, `controller`), links)
- Coordination patterns (goal decomposition trees (`voting`, `preferences`, `ballot`, ...), missions (`mAssistant`, `mVote`, `mController`))
- Rights and duties (norms)

- Including **organisation-reasoning** abilities into agents

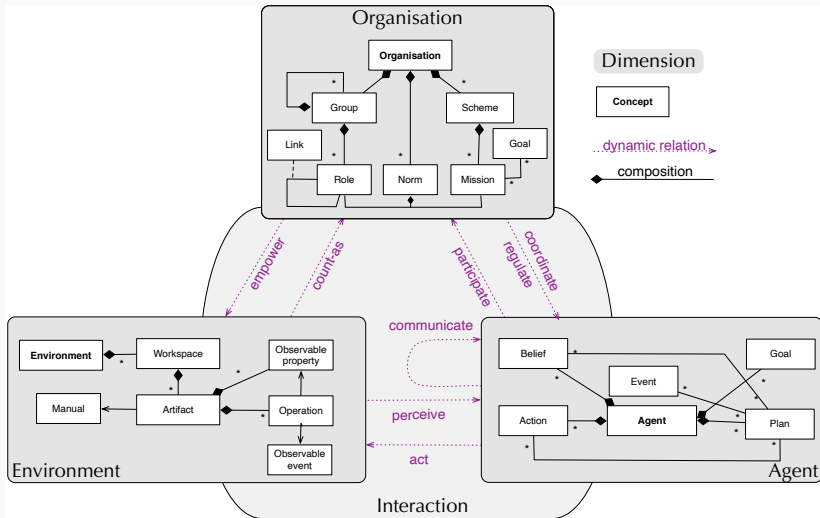
```
+play(Ag,assistant,GrId) <- .send(Ag,tell,hello).
```

```
+goalState(_,close_voting,_,_,satisfied) <- ...
```

- Including **norm-reasoning** abilities into agents

```
+obligation(Ag,Norm,achieved(_,Goal,_),Deadline)  
  : .my_name(Ag) & good(mood)  
  <- !Goal.
```

# Multi-Agent Oriented Programming: Global View



# Mainstream Application Domains and Technologies

- **ChatBot** [Engelmann et al., 2021]
- Mobile and **Wearable** Apps [Croatti and Ricci, 2021]
- IoT, IIoT, Web of Things and **Web** technologies: Hybrid Communities of Agents and Humans on the Web [Ciortea et al., 2019b, Ciortea et al., 2019a] (HyperAgent project – ANR-FSF Project)
- Robotics and **Drones** (embedded agents) [Menegol et al., 2018]
- Agent-Based Mixed Reality Environments in **Healthcare** [Croatti et al., 2020]
- Multi-Agent for **Industry of the Future** [Ciortea et al., 2018]
- Multi-Agent for Collaborative **Learning** [da Silveira Colissi et al., 2021]

- A Multi-Agent System is not only agents, or only organization, or only environment, or only interaction! It has all these dimensions! All are **first class entities**!
- MAOP proposes a **separation of concerns** between Agent, Environment, Interaction and Organization that brings rich features to engineer open, long lived, agile, non centralized, distributed intelligent systems
- MAOP paves the way to the **inclusion of physical, digital, human and social worlds** to define socio-cognitive, physical and digital systems
- **JaCaMo** proposes a seamless integration of these different abstractions interfacing to mainstream technologies



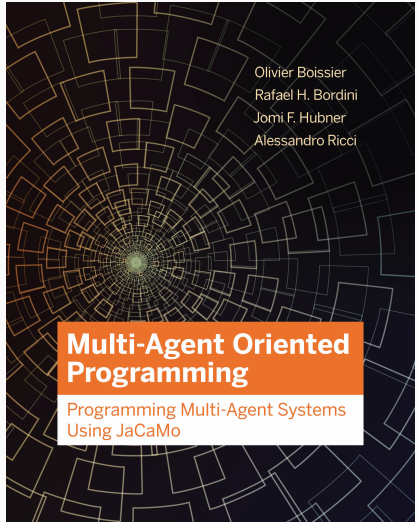
## Challenges and Some directions




- **Explainability** for all dimensions: going beyond the agents
- **Ethics** of intelligent system at the individual level but also considering the collective level w.r.t. organizational values, norms, cultures
- **Hybrid** communities of people and agents interconnecting physical and virtual environments, laws in the social world with laws in the digital world
- **Scalability** - distributed environment: agents on the Web, agents in industrial environments, in smart cities

# Acknowledgements

- Rafael Bordini (PUCRS, Brazil), Alessandro Ricci (UniBo, Italy)
- Jaime Sichman (USP Brazil), various colleagues and students
- JaCaMo users for helpful feedback
- CNPq, CAPES, ANP, ANR for supporting some of our current research
- Schloss Dagstuhl – Leibniz Center for Informatics

- <http://jacamo.sourceforge.net/>
- JaCaMo Book, MIT Press  
[Boissier et al., 2020]
- Jason Book, Wiley  
[Bordini et al., 2007]



-  Boissier, O., Bordini, R. H., Hübner, J., and Ricci, A. (2020).  
*Multi-Agent Oriented Programming: Programming Multi-Agent Systems Using JaCaMo.*  
MIT Press.
-  Bordini, R. H., Hübner, J. F., and Wooldrige, M. (2007).  
*Programming Multi-Agent Systems in AgentSpeak using Jason.*  
Wiley Series in Agent Technology. John Wiley & Sons.
-  Bratman, M. E. (1987).  
*Intention, Plans, and Practical Reason.*  
Harvard University Press, Cambridge.



Ciortea, A., Boissier, O., and Ricci, A. (2019a).

**Engineering world-wide multi-agent systems with hypermedia.**

In Weyns, D., Mascardi, V., and Ricci, A., editors, *Engineering Multi-Agent Systems - 6th International Workshop, EMAS 2018, Stockholm, Sweden, July 14–15, 2018, revised selected papers*, volume 11375 of *LNCS*, pages 285–301. Springer.



Ciortea, A., Mayer, S., Gandon, F. L., Boissier, O., Ricci, A., and Zimmermann, A. (2019b).

**A decade in hindsight: The missing bridge between multi-agent systems and the world wide web.**

In Elkind, E., Veloso, M., Agmon, N., and Taylor, M. E., editors, *Proceedings of the 18th International Conference on Autonomous Agents*

*and MultiAgent Systems, AAMAS'19, Montreal, Canada, May 13–17, 2019*, pages 1659–1663.



Ciortea, A., Mayer, S., and Michahelles, F. (2018).

**Repurposing manufacturing lines on the fly with multi-agent systems for the web of things.**

In André, E., Koenig, S., Dastani, M., and Sukthankar, G., editors, *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10–15, 2018*, pages 813–822. International Foundation for Autonomous Agents and Multiagent Systems.



Cohen, P. R. and Levesque, H. J. (1987).

**Intention = choice + commitment.**

In *Proceedings of the 6th National Conference on Artificial Intelligence*, pages 410–415. Morgan Kaufmann.



Croatti, A., Bottazzi, M., and Ricci, A. (2020).

**Agent-based mixed reality environments in healthcare: The smart shock room project.**

In Demazeau, Y., Holvoet, T., Corchado, J. M., and Costantini, S., editors, *Advances in practical applications of agents, multi-agent systems, and trustworthiness. The PAAMS collection*, pages 398–402, Cham. Springer International Publishing.



Croatti, A. and Ricci, A. (2021).

**Programming agent-based mobile apps: The jaca-android framework.**

In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '21*, page 1724–1726, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.



da Silveira Colissi, M., Vieira, R., Mascardi, V., and Bordini, R. H. (2021).

**A chatbot that uses a multi-agent organization to support collaborative learning.**



In Stephanidis, C., Antona, M., and Ntoa, S., editors, *HCI international 2021 - posters*, pages 31–38, Cham. Springer International Publishing.



Engelmann, D., Damasio, J., Krausburg, T., Borges, O., Colissi, M., Panisson, A. R., and Bordini, R. H. (2021).

**Dial4jaca—a communication interface between multi-agent systems and chatbots.**

In *International conference on practical applications of agents and multi-agent systems*, pages 77–88. Springer.



Menegol, M. S., Hübner, J. F., and Becker, L. B. (2018).

**Coordinated UAV search and rescue application with JaCaMo.**

In Demazeau, Y., Bajo, B. A., and Fernández-Caballero, A., editors, *Proc. of 16th International Conference International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2018)*, volume 10978 of *LNCS*, pages 335–338.



Rao, A. S. and Georgeff, M. P. (1995).

**BDI agents: from theory to practice.**

In Lesser, V., editor, *Proceedings of the First International Conference on MultiAgent Systems (ICMAS'95)*, pages 312–319. AAAI Press.