UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD www.multiagent.fr

From Multiagent Simulation to Virtual Environment Simulation

Dr Stéphane GALLAND

Multiagent System & Simulation Group
Laboratoire Systèmes and Transports (IRTES-SET)
Institut de Recherche Transport Energie et Société (IRTES)
Université de Technologie de Belfort-Montbéliard (UTBM)
France

http://www.multiagent.fr/People:Galland_stephane
stephane.galland@utbm.fr

utbm

SET
Laboratoire Systèmes et Transports

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

www.multiagent.fr

From Multiagent Simulation to Virtual Environment Simulation

Stéphane GALLAND – stephane.galland@utbm.fr

SET Laboratoire Systèmes et Transports

utbm

version 2012.01

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

www.multiagent.fr

From Multiagent Simulation to Virtual Environment Simulation

Stéphane GALLAND – stephane.galland@utbm.fr

Simulation <

Multiagent-based Simulation <

From Multiagent-based Simulation to Virtual Environment Simulation <

Simulation Principles and Architectures <

Environment Model <

Motion Model <

Demos <

Laboratoire Systèmes et Transports

version 2012.01

An Overview of Simulation

- **Domains:** Multiagent-based simulation, Multiagent systems
- **Main goal:** study of complex systems
- **Why use of simulation ?**
  Simulation may be considered as a propert approach for studying systems that cannot be directly observed, measured or easily understood [Conte and Gilbert, 1995]

Some Definitions of Simulation

[Shannon, 1977]

The process of designing a model of a real system and conducting experiments with this model for the purpose either of understanding the behavior of the system or of evaluating various strategies (within the limits imposed by a criterion or a set of criteria) for the operation of the system.

Why simulate?

- Understand / optimize a system
- Scenarii/strategies evaluation, testing hypotheses to explain a phenomenom (decision-helping tool)
- Predicting the evolution of a system, eg. metrology

Some Definitions of Simulation

[Fishwick, 1997]
Computer simulation is the discipline of designing a model of an actual or theoritical physical system, executing the model on a digital computer, and analyzing the execution output.

Three fundamental tasks in all kinds of simulation

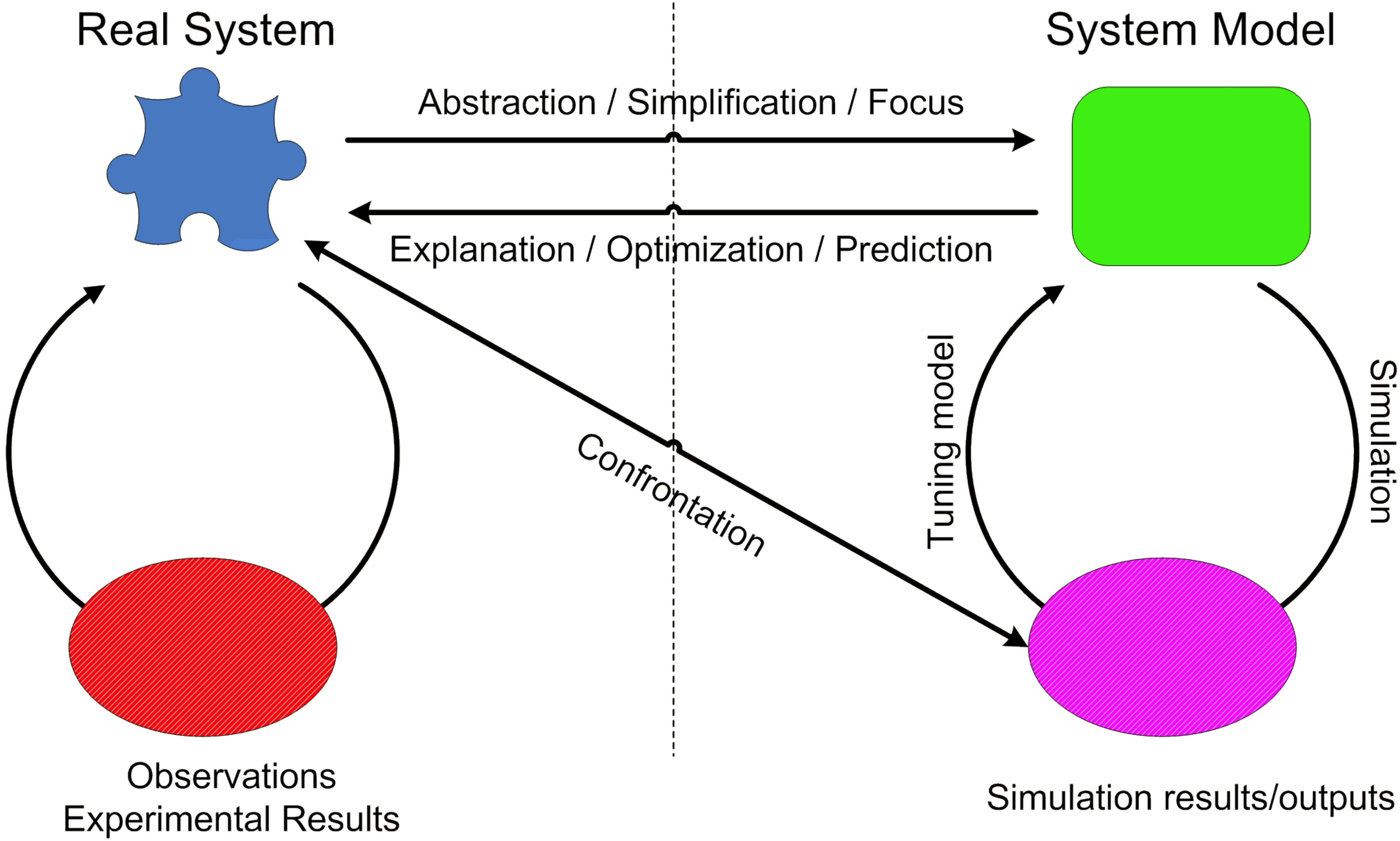
- Model design
- Model execution
- Result analysis

Simulation Process

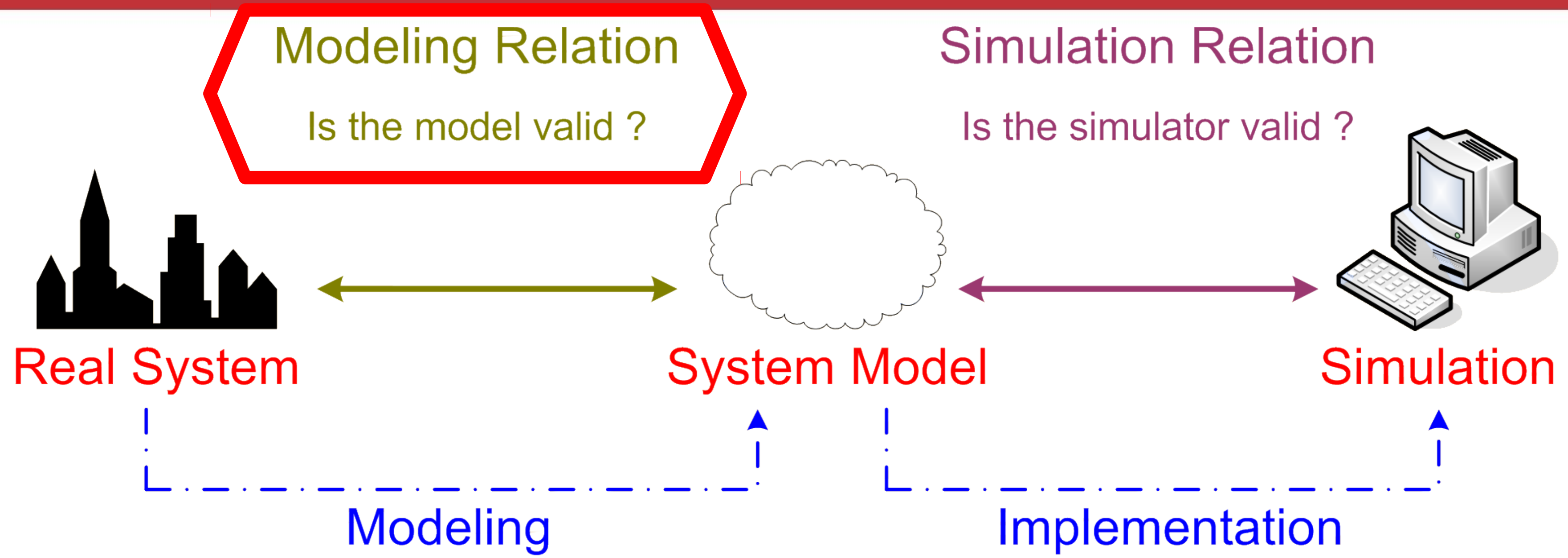
1) Requirement analysis: defining the objectives addressed by the model
2) Model conceptualization and specification
3) Implementation
4) Calibration and bug fixing
5) Experiments
6) Analysis of results

- Parallel activity : Model validation (against the real system)
- Model verification (end of each activity)

Simulation Basics

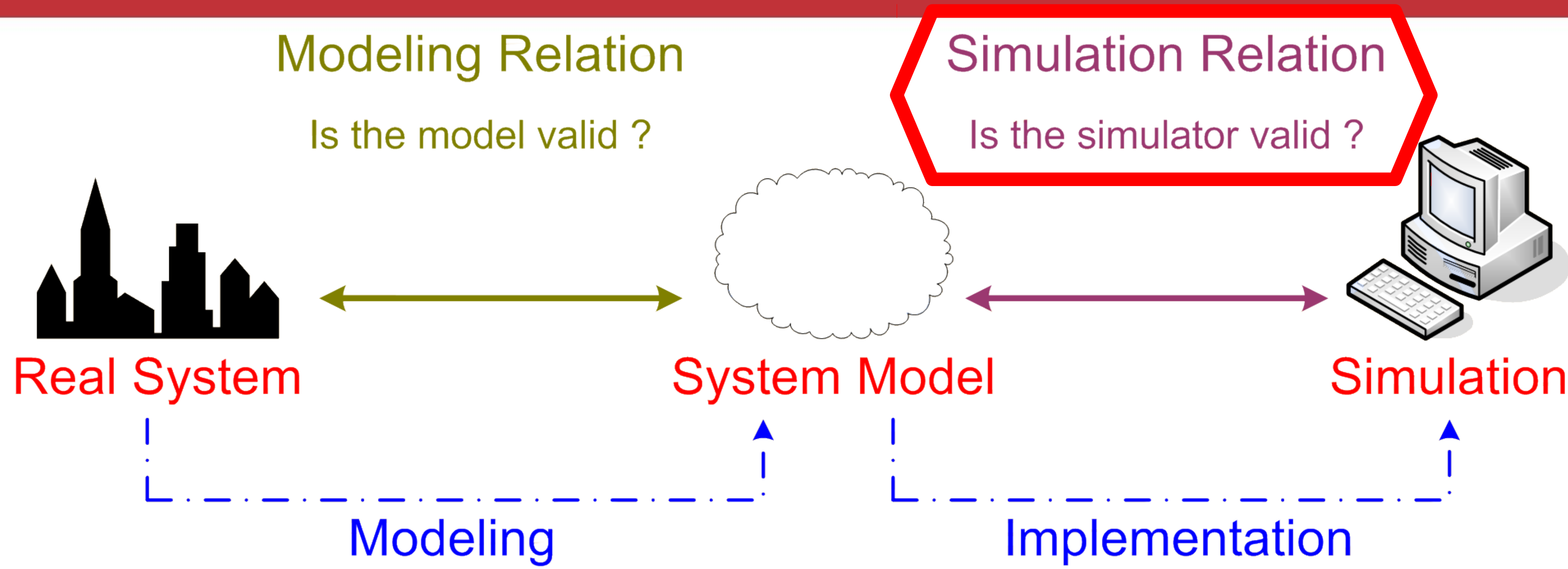


Modeling Relation : System-Model Relation



- To determine if the system model is an acceptable simplification in terms of quality criteria and experimentation objectives
- This relationship is directly related to the consistency of the model simulation
- [Zeigler et al., 2000]

Modeling Relation : System-Model Relation

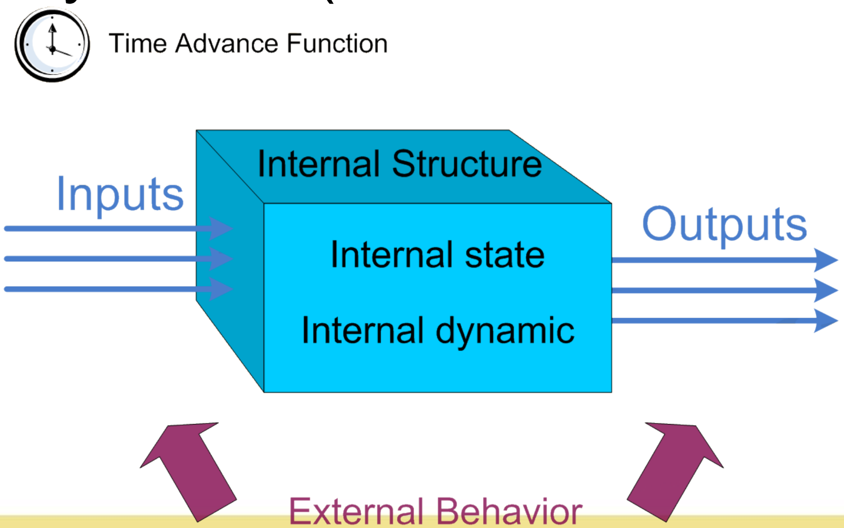


- To obtain a guarantee that the simulator, used to implement the model, correctly generates the behavior of the model.
- To be sure that the simulator reproduces clearly the mechanisms of change of state that are formalized in the model.
- [Zeigler et al., 2000]

What are we simulating?

A dynamic system

- See Systemic Theory, General System Theory
- Two fundamental aspects in a dynamic system :
  - The external behavior of the system (at its bounds) : the observable reactions of the system from outside it.
  - The internal structure of the system : its internal state and its inner dynamics (state-transition function).



Dynamic of System : Time Advance Function

- Primarily defined according the way it evolves over time.
- One of the most important characteristics of a model :

How the passage of time is represented

- **3 main approaches** :
  - Continous Model: state variables evolve continously. Differential equation system specification.
  - Discrete Model: time axis is discretized following a constant period Δt (the time step)
  - Event-based Model : state variables evolve discretely at specific instants represented by events.

Dynamic of System : Time Advance Function

- Primarily defined according the way it evolves over time.
- One of the most important characteristics of a model :

How the passage of time is represented

- **3 main approaches** :
  - Continous Model: state variables evolve continously. Differential equation system specification.

Dynamic of System : Time Advance Function

- Primarily defined according the way it evolves over time.
- One of the most important characteristics of a model :

> How the passage of time is represented

- **3 main approaches** :
  - Continous Model: state variables evolve continously. Differential equation system specification.
  - Discrete Model: time axis is discretized following a constant period Δt (the time step)

$$\sigma(t+\Delta t)=\phi(\sigma(t))$$

With $\sigma(t)$ the state of the system at t and ϕ the transition function.

Dynamic of System : Time Advance Function

- Primarily defined according the way it evolves over time.
- One of the most important characteristics of a model :

> How the passage of time is represented

- **3 main approaches** :
  - Continous Model: state variables evolve continously. Differential equation system specification.
  - Discrete Model: time axis is discretized following a constant period Δt (the time step)
  - Event-based Model : state variables evolve discretely at specific instants represented by events.

Classical Topology of the Simulation

- Typology according to the granularity of the simulation, the level of detail that is possible in the model.
- **Micro Simulation**
  - explicitly attempts to model the behaviors of each individual.
  - The system structure is viewed as emergent from the interactions between the individuals.

[Hoogendoorn and Bovy, 2001, Davidsson, 2000]

Classical Topology of the Simulation

- Typology according to the granularity of the simulation, the level of detail that is possible in the model.
- Micro Simulation
- **Macro Simulation**

  - based on mathematical models, where the characteristics of a population are averaged together.
  - Simulate changes in these averaged characteristics for the whole population.
  - The set of individuals is viewed as a structure that can be characterized by a number of variables

[Hoogendoorn and Bovy, 2001, Davidsson, 2000]

Classical Topology of the Simulation

- Typology according to the granularity of the simulation, the level of detail that is possible in the model.
- Micro Simulation
- Macro Simulation
- **Meso Simulation**
  - Something in between

[Hoogendoorn and Bovy, 2001, Davidsson, 2000]

Classical Topology of the Simulation

- Typology according to the granularity of the simulation, the level of detail that is possible in the model.
- Micro Simulation
- Macro Simulation
- Meso Simulation
- **Multi-Level Simulation**
  - combines various levels (micro-macro for example)

> Multiagent-Based Simulation (MABS) is traditionnally considered as a special form of micro simulation

[Hoogendoorn and Bovy, 2001, Davidsson, 2000]

Traditional Approaches

- Equation-based model,
  - mainly from computational physics : particle system evolution, Vortex methods, molecular dynamics, fluid dynamics, finite element.
- Process-oriented Models
  - Queuing models/networks
  - Petri-nets
- Object-oriented simulation
- Cellular automata

Macro Simulation : Main Characteristics and Advantages

- Usually are : equation-based model, differential equations
  - Well understood, established mathematical framework.
- Formulas are concise and form a complete model
- Proven success in several domains
- Usually, the volume and the accuracy of data required for the initialization are much smaller than for other approaches
- Usually, lighter in terms of computations.
- Easier to simulate large scale systems

Macro Simulation : Limitations and Drawbacks (1/2)

- Difficult to switch from micro to macro level (transition)
- Difficult (or impossible) to represent certain behaviors, eg. predation, mating rituals, acquisition of food...
- Does not represent behaviors but the results/outputs of the behaviors (aggregated data: number of descendants, food intake quantity...)

Macro Simulation : Limitations and Drawbacks (2/2)

- Not appropriate for certains kinds of system :
  - Systems that draw their dynamics from flexible local interactions
  - Social systems, social hierarchies
  - Emergent phenomena and self-organizing systems : biological systems, trafic systems.
  - Multi-level systems,
  - Intelligent human behavior

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD www.multiagent.fr

From Multiagent Simulation to Virtual Environment Simulation

Stéphane GALLAND – stephane.galland@utbm.fr

Laboratoire Systèmes et Transports

version 2012.01

MABS: Main Characteristics and Advantages

- More flexible than macroscopic models to simulate spatial and evolutionary phenomena.
- Dealing with real multiagent systems directly : real Agent = simulated Agent
- Allows modelling of adaptation and evolution
- Heterogeneous space and population
- Multi-level modeling : integrate different levels of observation, and of agent's behaviors.

MABS: Limitations and Drawbacks

- Offer a significant level of accuracy at the expense of a larger computational cost.
- Require many and accurate data for their initialization
- It is difficult to apply to large scale systems.
- Actual simulation models are costly in time and effort.

MABS: General Idea

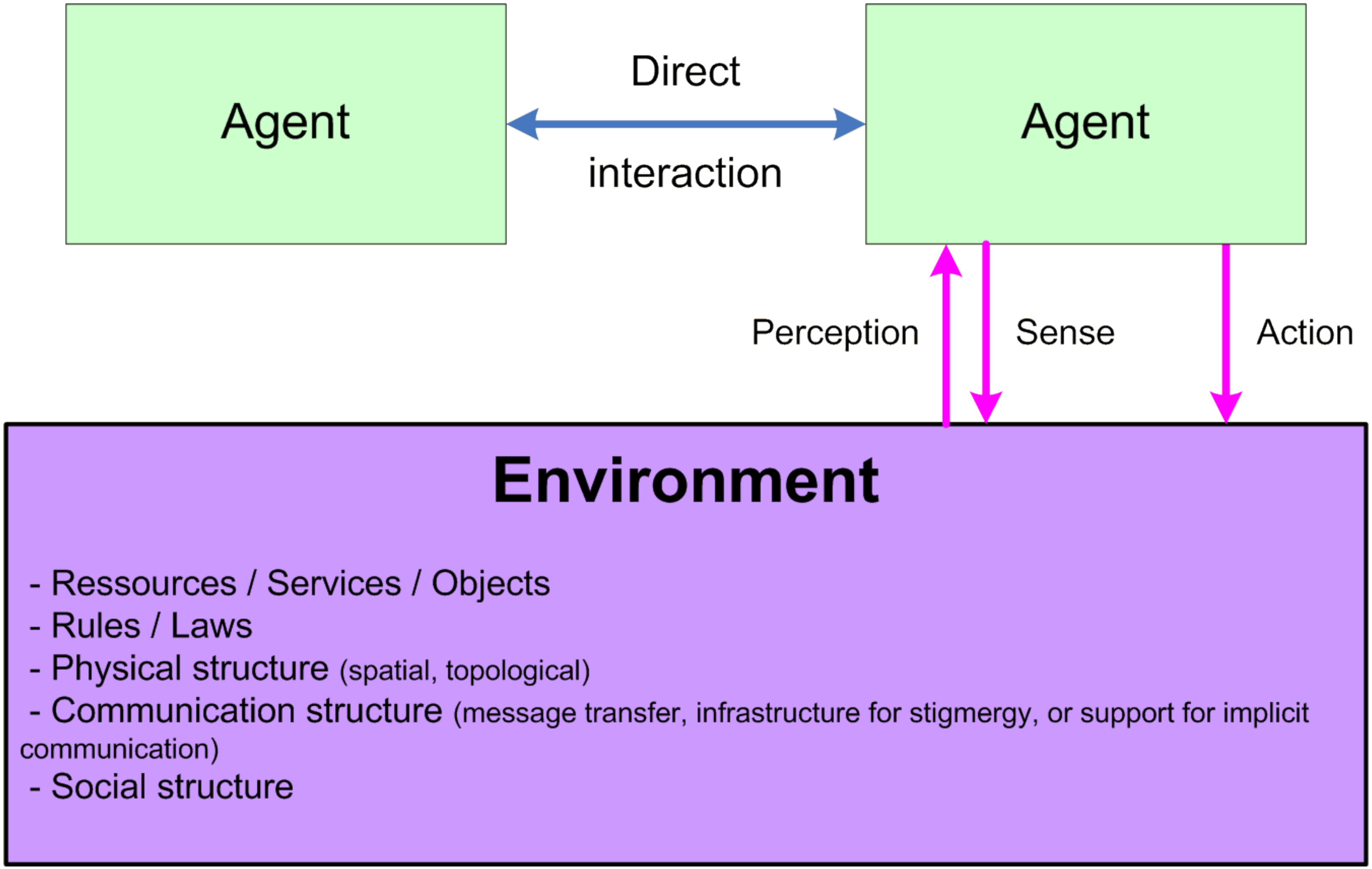
- Create an artificial world composed of interacting agents.
- The behavior of an agent results from :
  - its perceptions/observations
  - its internal motivations/goals/beliefs/desires
  - its eventual representations
  - its interaction with the environment (indirect interactions, ressources) and the other agents (communications, direct interactions, stimuli)
- Agents act and modify the state of their environment through their actions.
- We observe the results of their interactions like in a Virtual Lab.

Fundamental Properties of an Agent

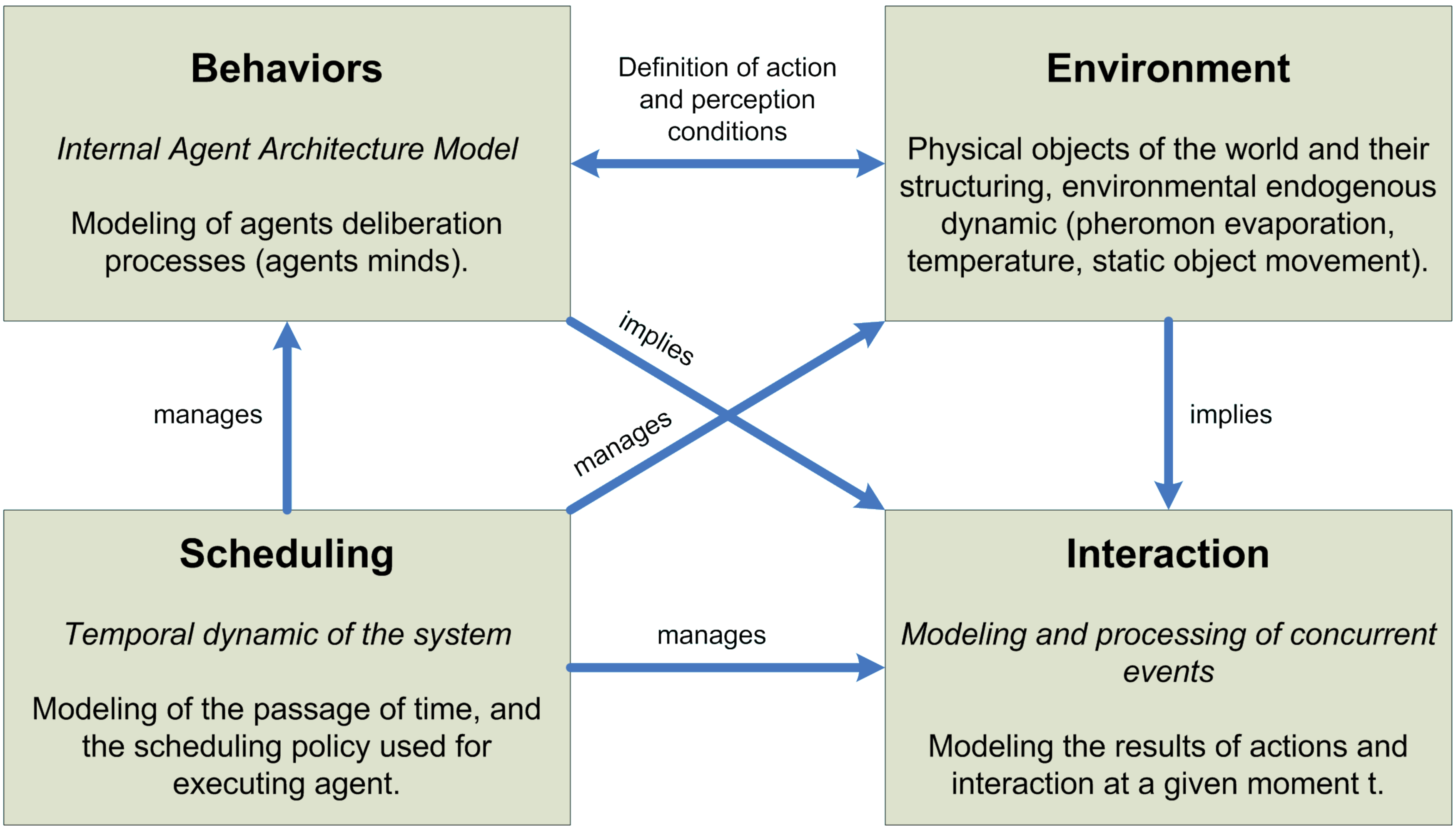
- **Autonomy** owns a internal state on which it has a total control, inaccessible to other agents. Takes decisions according to this internal state without any outside intervention (human or other agents)
- **Reactivity** evolves in an environment, perceives it and reacts to changes that occur (especially through their actions)
- **Social** is a social entity able to interact with other agents directly or via the environment
- **Proactivity** does not only react to its environment but is also able of producing its own actions motivated by its personal goals/motivations.

[Wooldridge and Jennings, 1995]

MABS: an Overview



MABS: Designing a multiagent simulation model



[Michel, 2004]

An Execution Platform : Janus

JANUS PROJECT

The Janus Project
Holonic Multi Agent Platform

- **Janus is an open-source multi-agent platform** fully implemented in Java 1.6.
- Janus enables developers to quickly create web, enterprise and desktop multiagent-based applications.
- Janus is free for non-commercial use and distributed under the terms of the GPLv3.
- Website : http://www.janus-project.org
- Janus is jointly developed by the multiagent teams of the **Laboratoire Systèmes et Transports** and the **Centro de Investigación de Tecnologías Avanzadas de Tucumán**.

MULTIAGENT GROUP
SET Laboratoire Systèmes et Transports
CITAT

Janus

JANUS PROJECT

The Janus Project
Holonic Multi Agent Platform

- Agent and organizational platform
- Agent-agent communications
- Role-role communications
- Agent-role communications
- Multi-thread and execution policies
- Network support (JXTA peer-to-peer)
- Agent observation toolkit
- OSGi and Maven compliant
- BDI, Language Acts, Android
- Environment Model : Jaak extension, or JaSim

- http://www.janus-project.org

Jade

- Agent platform
- Agent-agent communications
- Multi-thread and execution policies
- Network support
- Agent observation toolkit
- BDI, Language Acts

- Environment Model : not directly included

- http://jade.tilab.com

compliant

NetLogo

- Multi-agent platform
- Agent = turtle
- Turtle communication : direct and stigmergy
- Multi-thread and execution policies
- Observation toolkit

- Environment Model : embedded in NetLogo

- http://ccl.northwestern.edu/netlogo

Swarm

- A-Life platform
- Agent = swarm
- Swarm communication : direct and stigmergy
- Multi-thread and execution policies
- Swarm observation toolkit

- Environment Model : embedded in Swarm

- http://www.swarm.org

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD
www.multiagent.fr

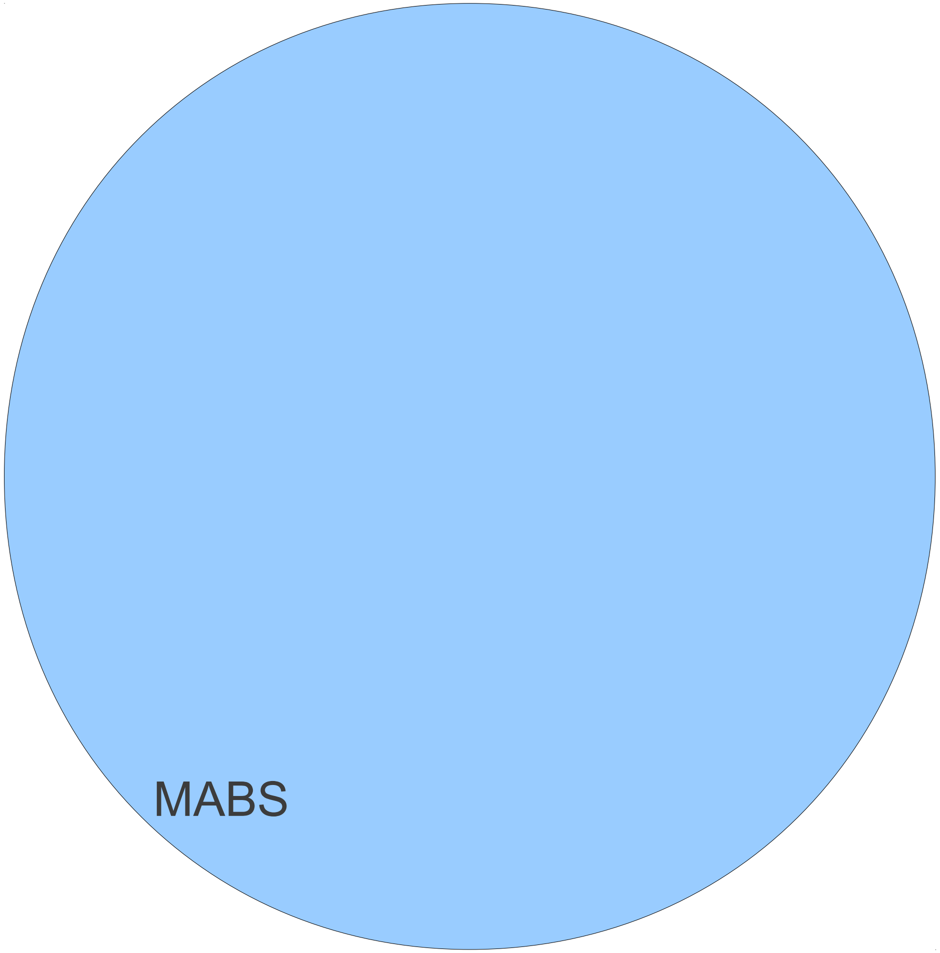
From Multiagent Simulation to Virtual Environment Simulation

Stéphane GALLAND – stephane.galland@utbm.fr

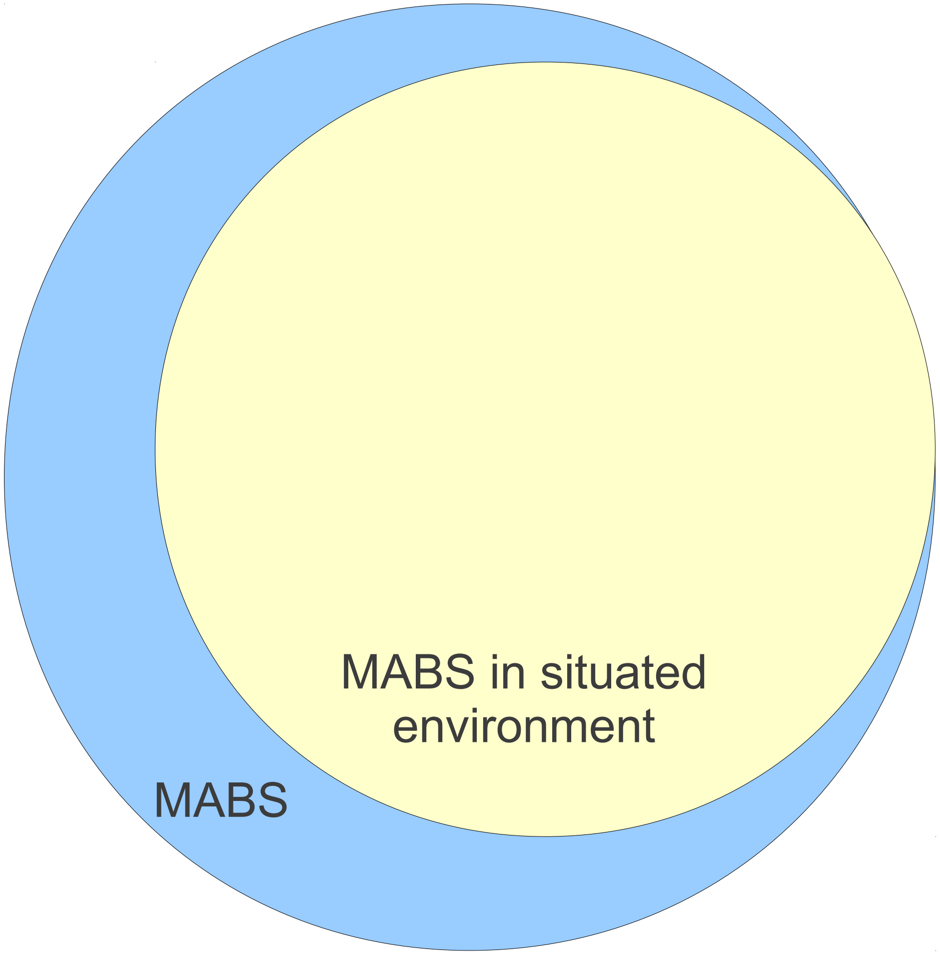
Laboratoire Systèmes et Transports

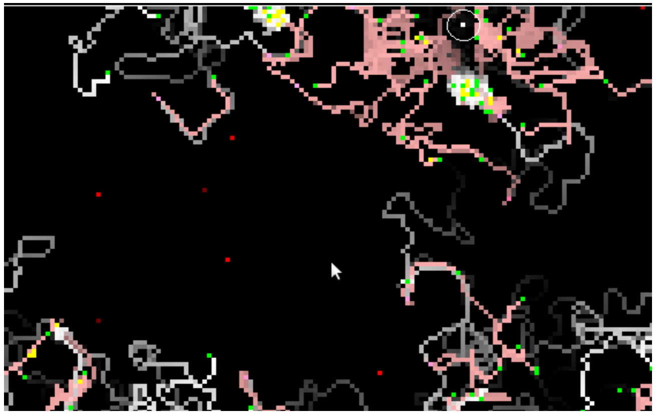
version 2012.01

MABS and Virtual Life Simulation



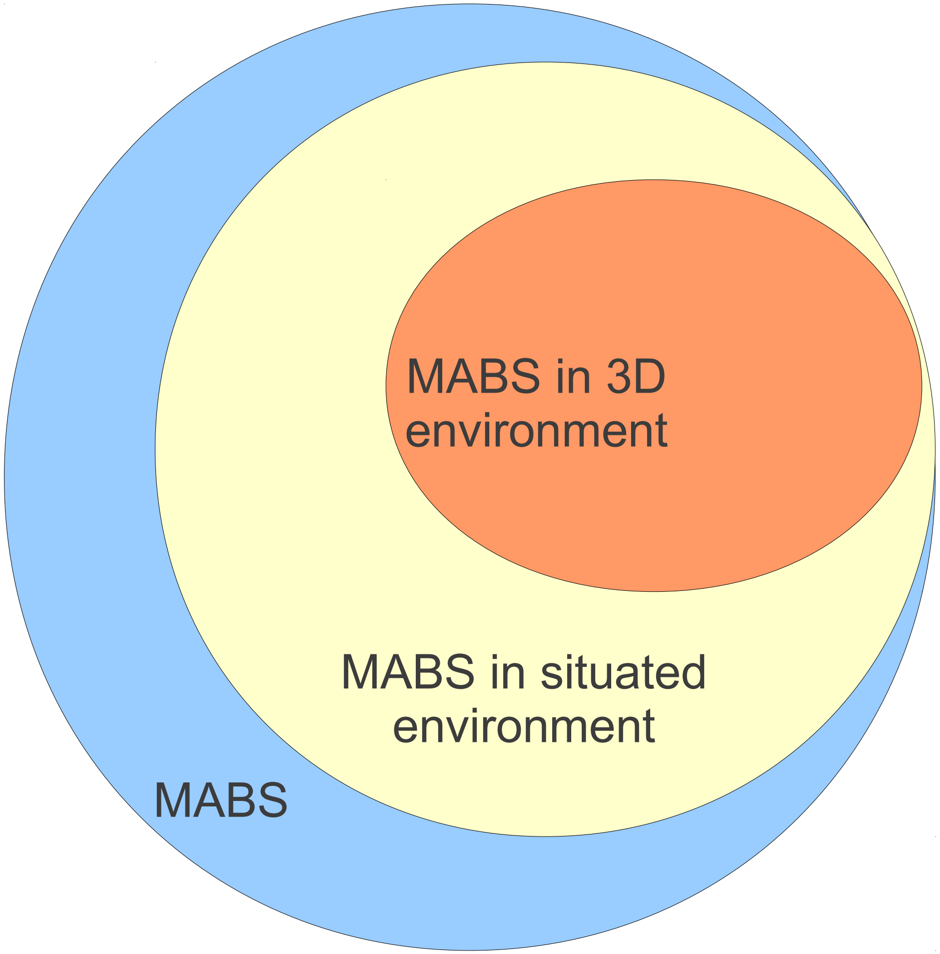
MABS and Virtual Life Simulation





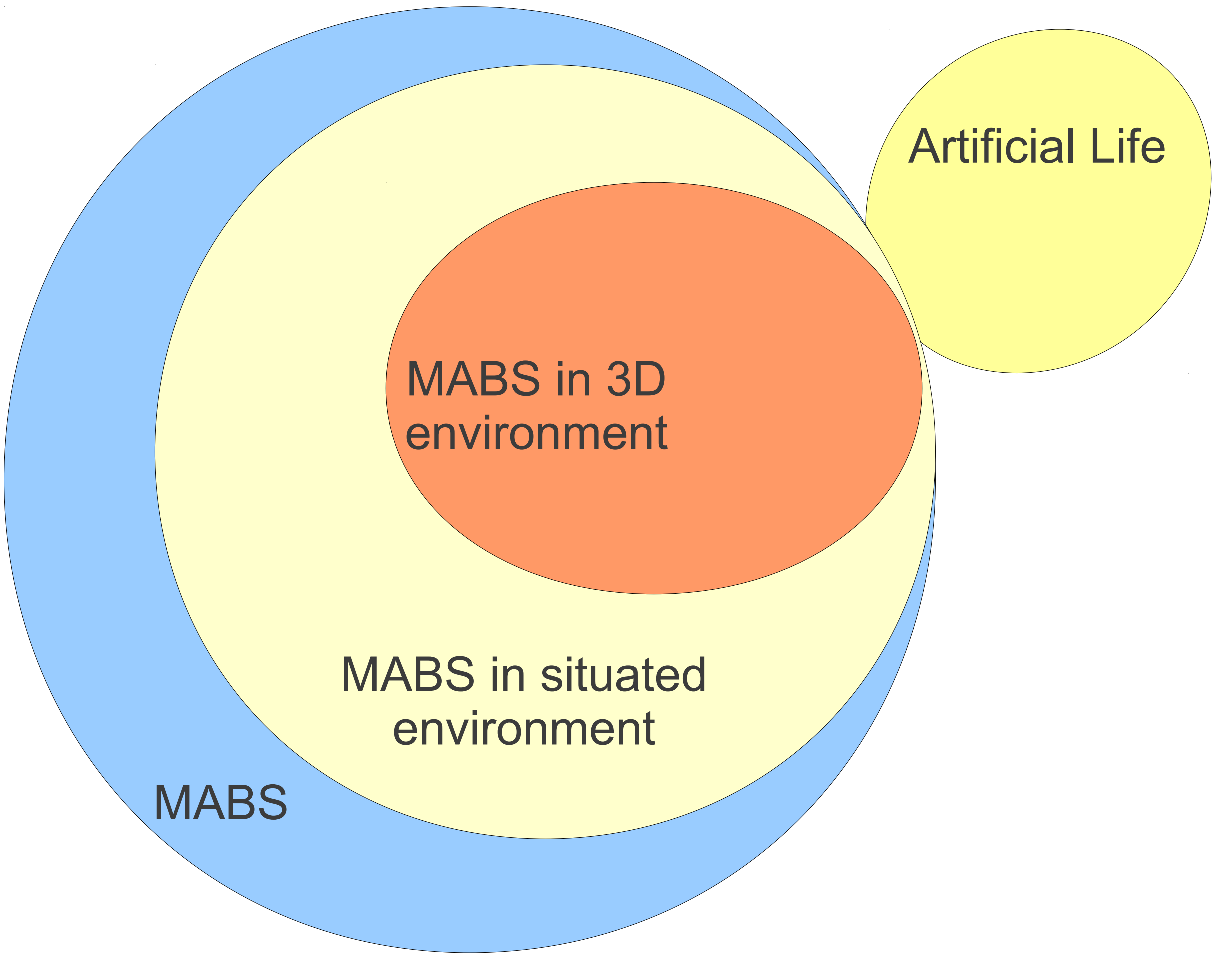
Ant Colony Simulation

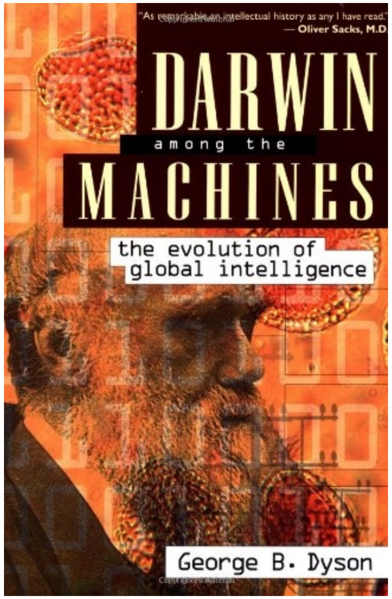
MABS and Virtual Life Simulation



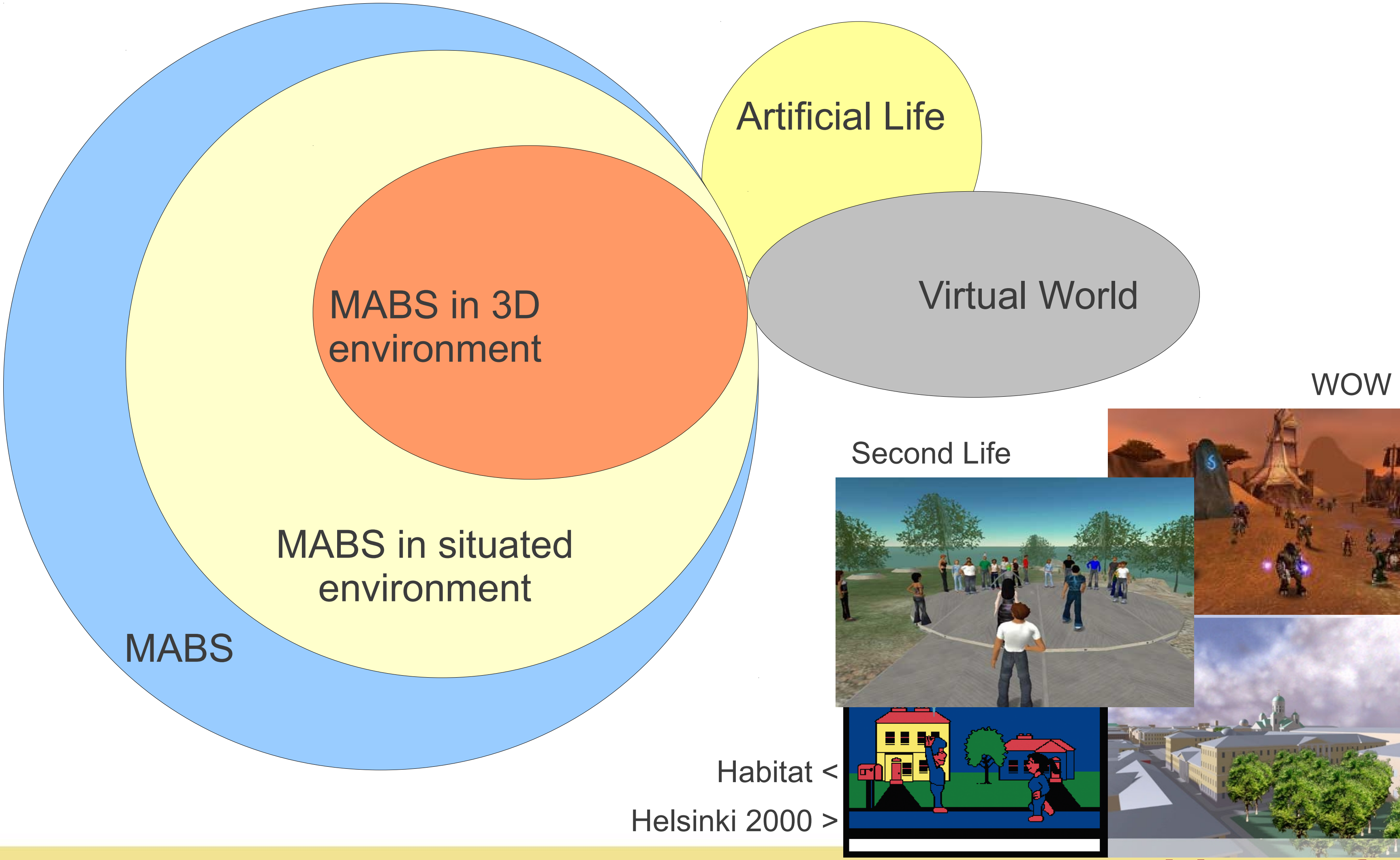
Full Spectrum Warrior (2004)

MABS and Virtual Life Simulation





MABS and Virtual Life Simulation



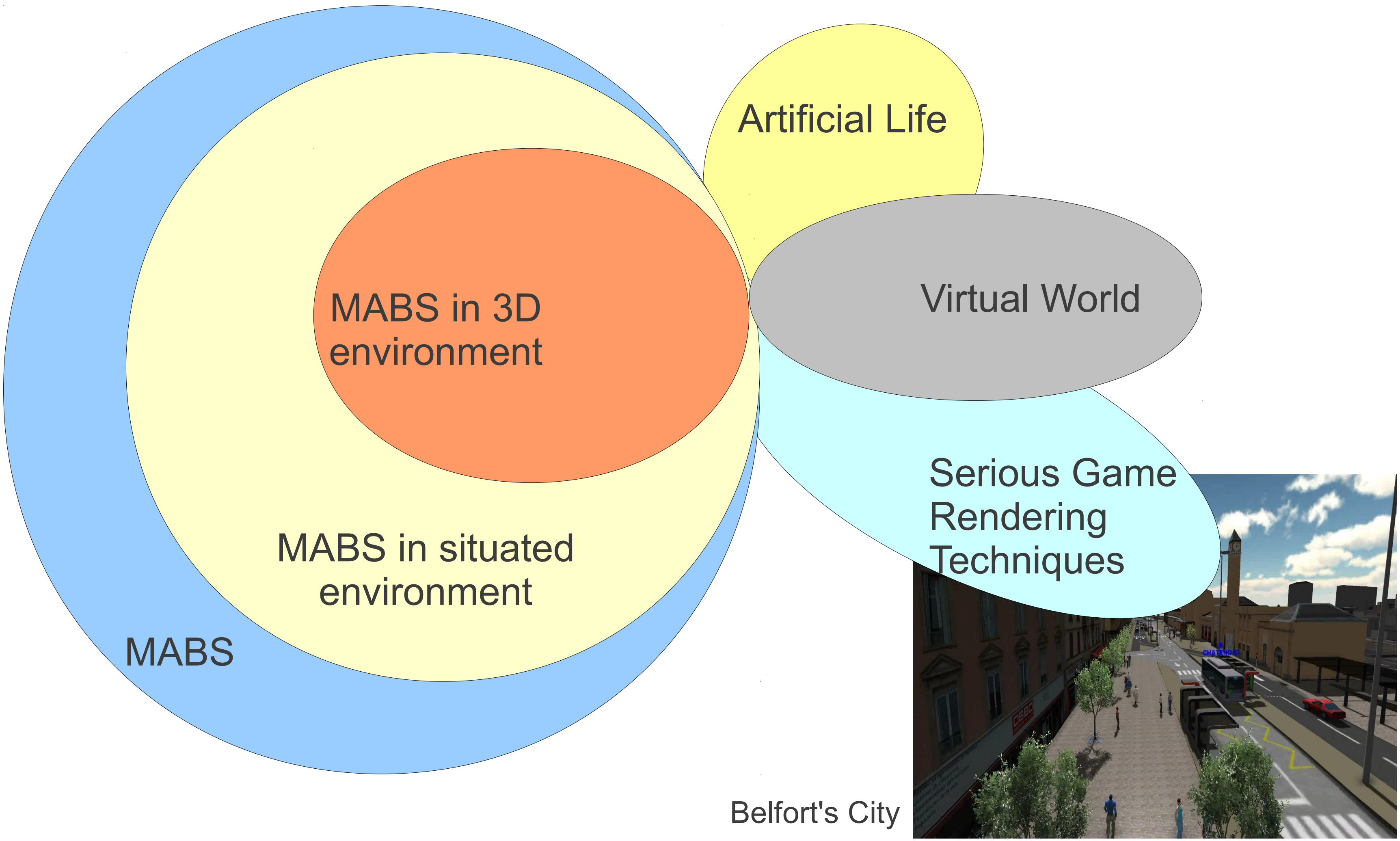
WOW

Second Life

Habitat <

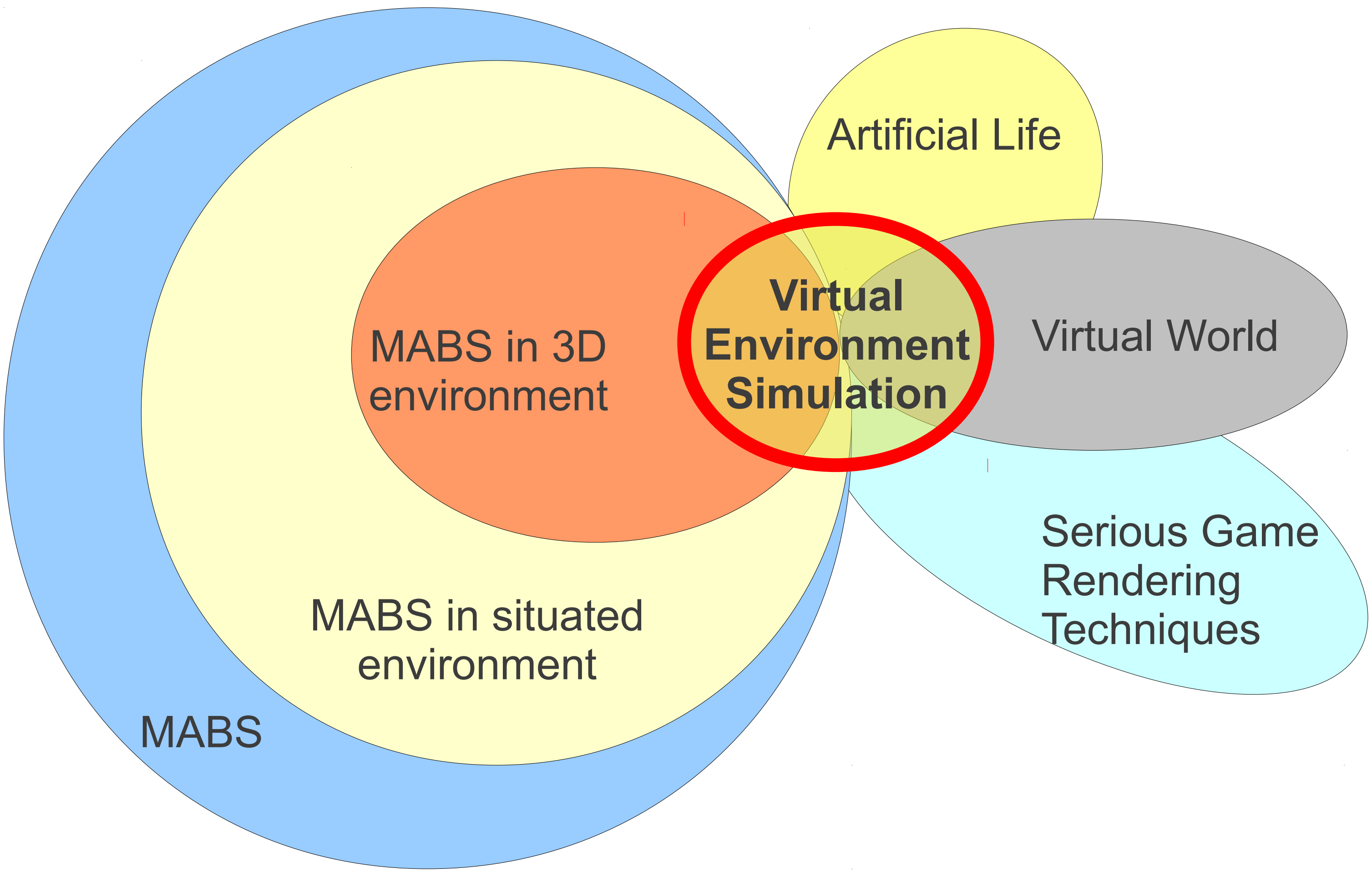
Helsinki 2000 >

MABS and Virtual Life Simulation



Belfort's City

MABS and Virtual Life Simulation



UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD www.multiagent.fr

From Multiagent Simulation to Virtual Environment Simulation

Stéphane GALLAND – stephane.galland@utbm.fr

Simulation <

Multiagent-based Simulation <

From Multiagent-based Simulation to Virtual Environment Simulation <

Simulation Principles and Architectures <

Environment Model <

Motion Model <

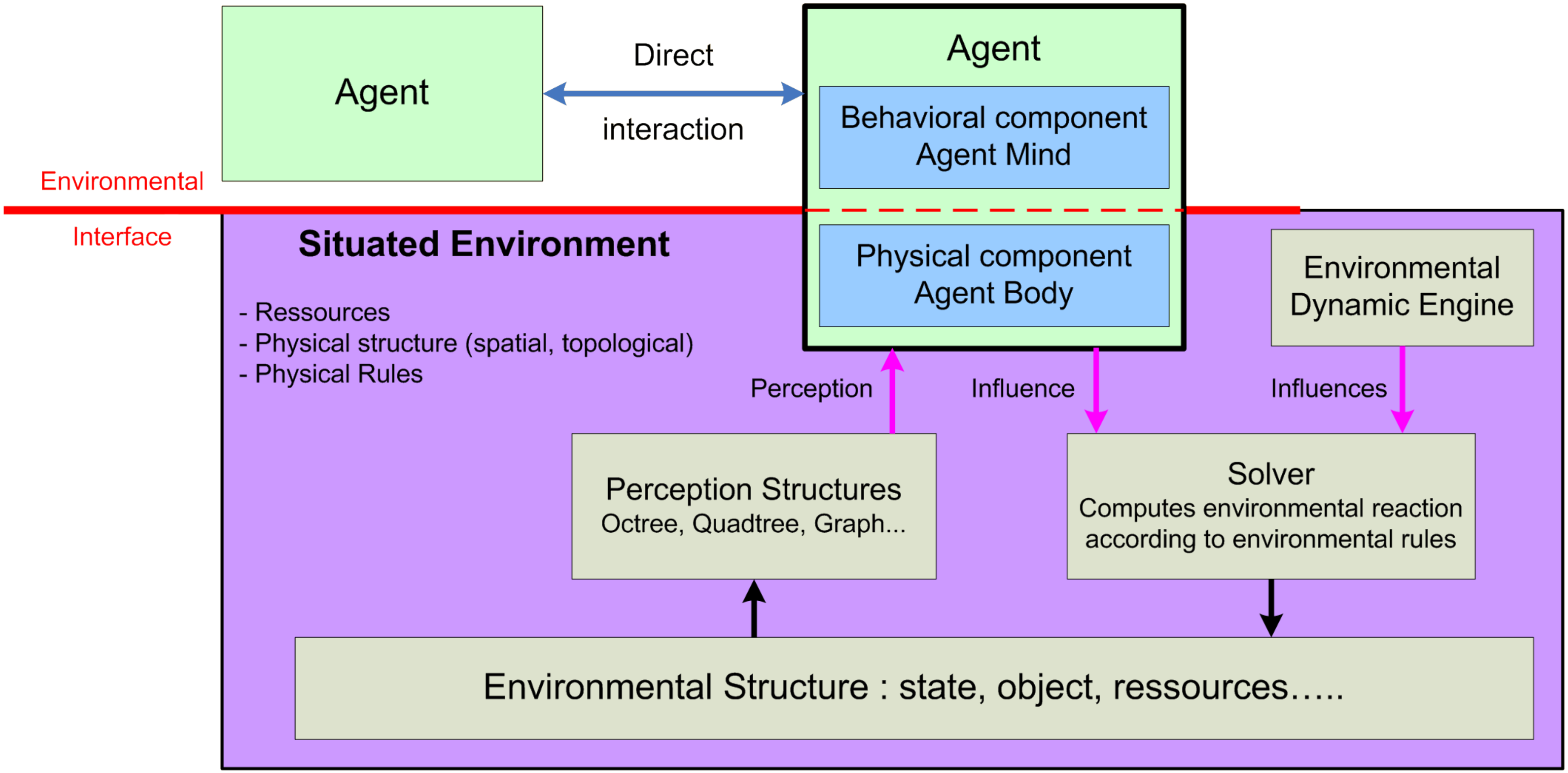
Demos <

SET Laboratoire Systèmes et Transports

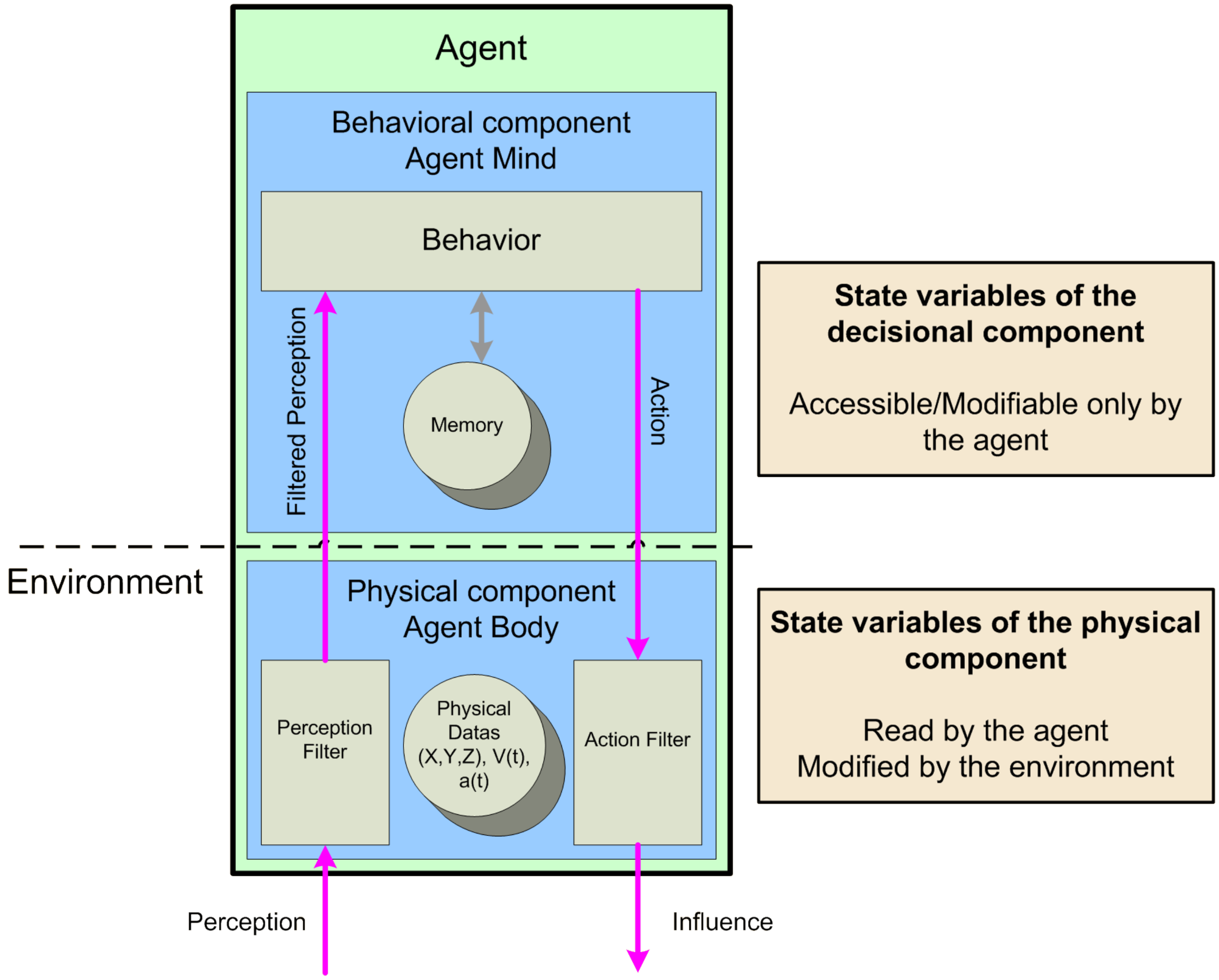
utbm

version 2012.01

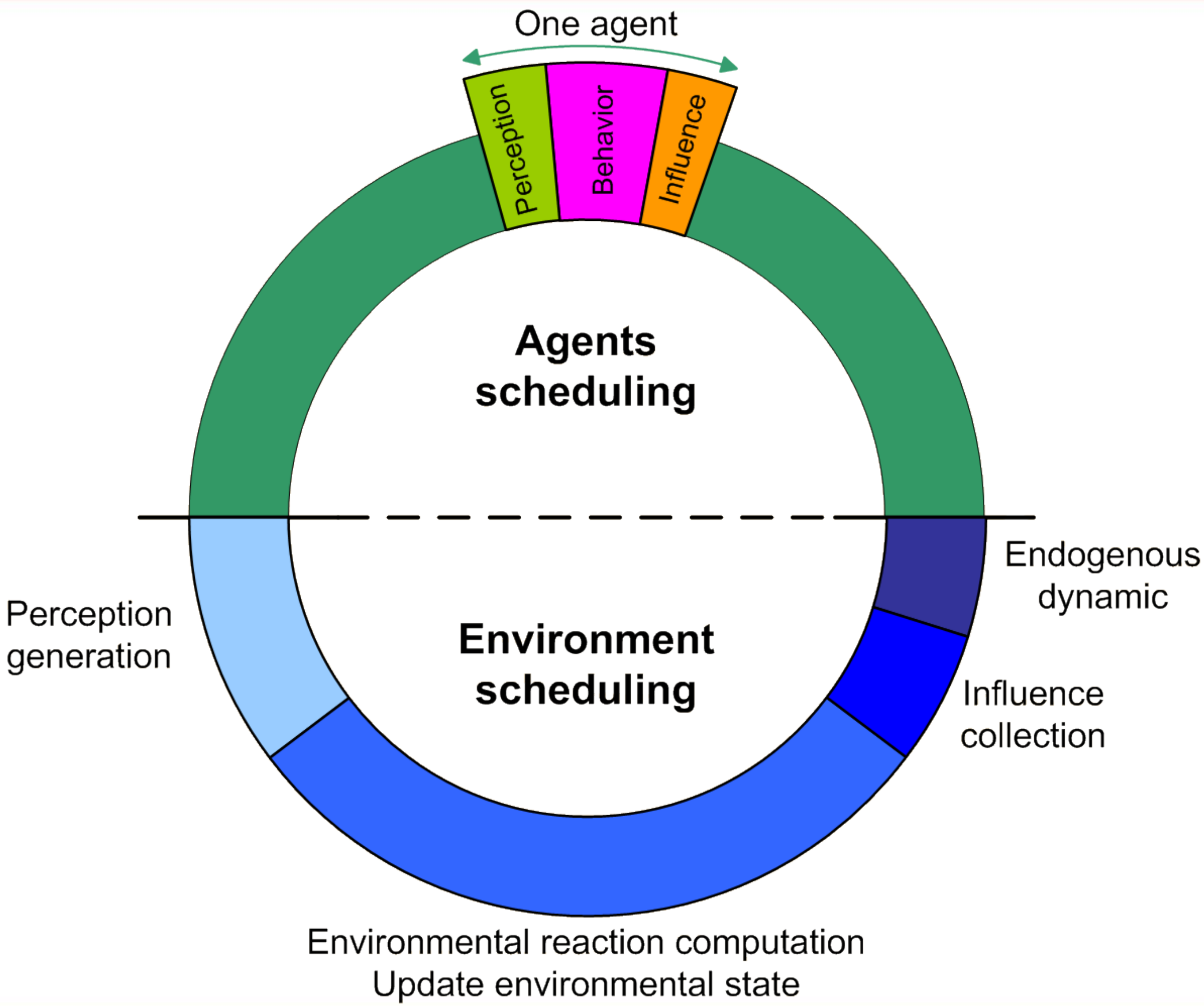
Classical Architecture : Environment Model



Classical Architecture : Body/Mind Distinction



Classical Architecture : Scheduling Model



From Multiagent Simulation to Virtual Environment Simulation

Stéphane GALLAND – stephane.galland@utbm.fr

SET Laboratoire Systèmes et Transports

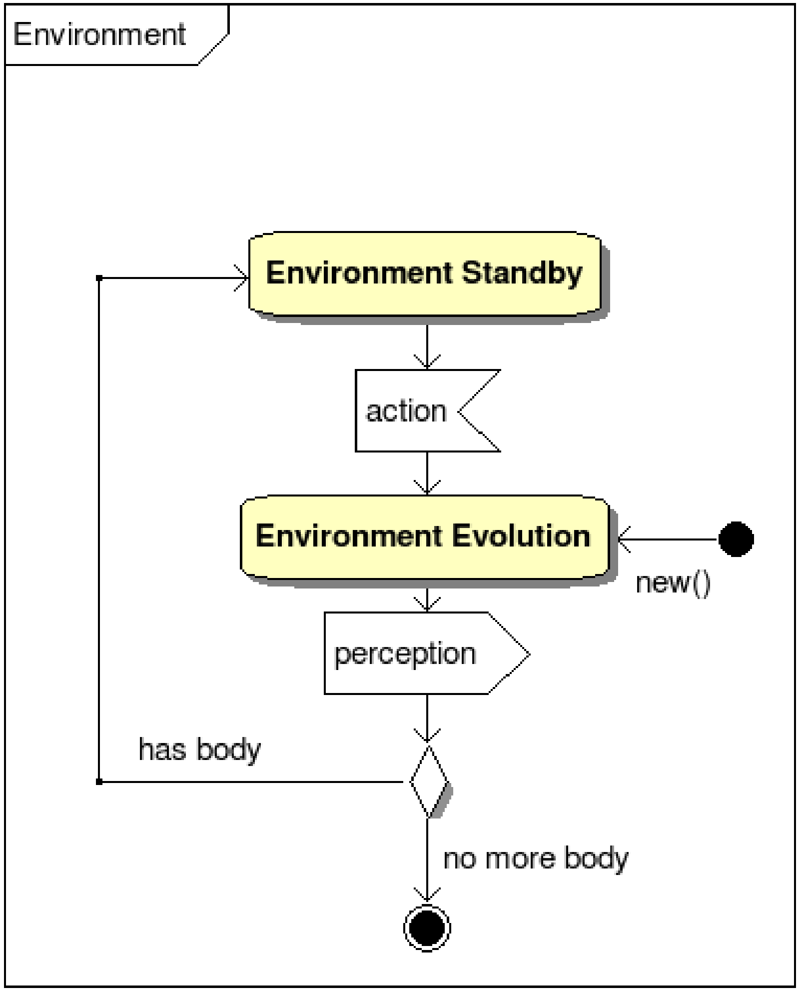
utbm

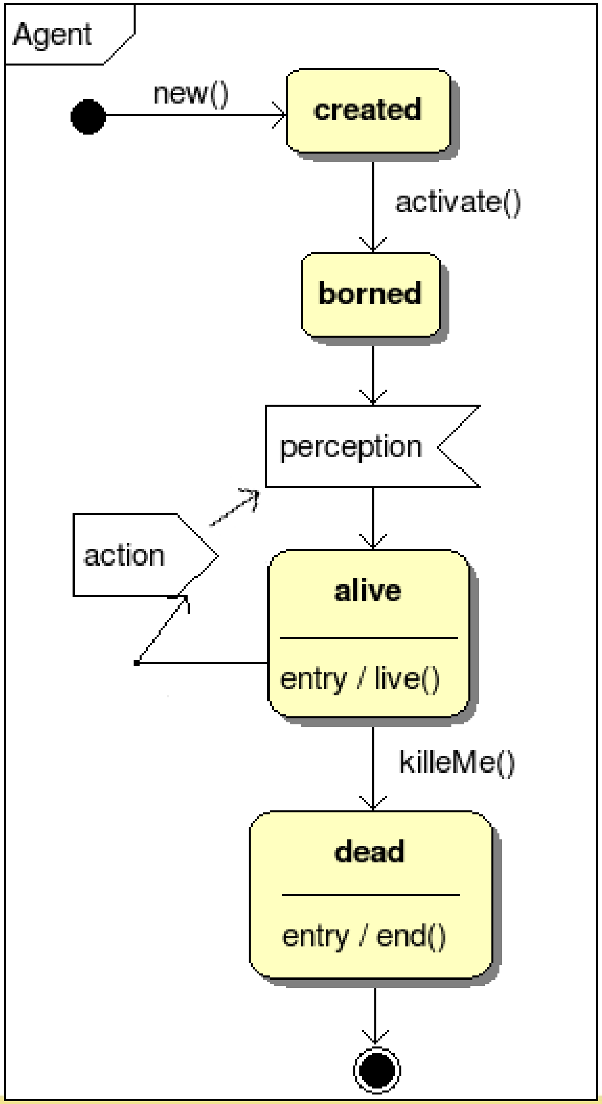
version 2012.01

Environment Content

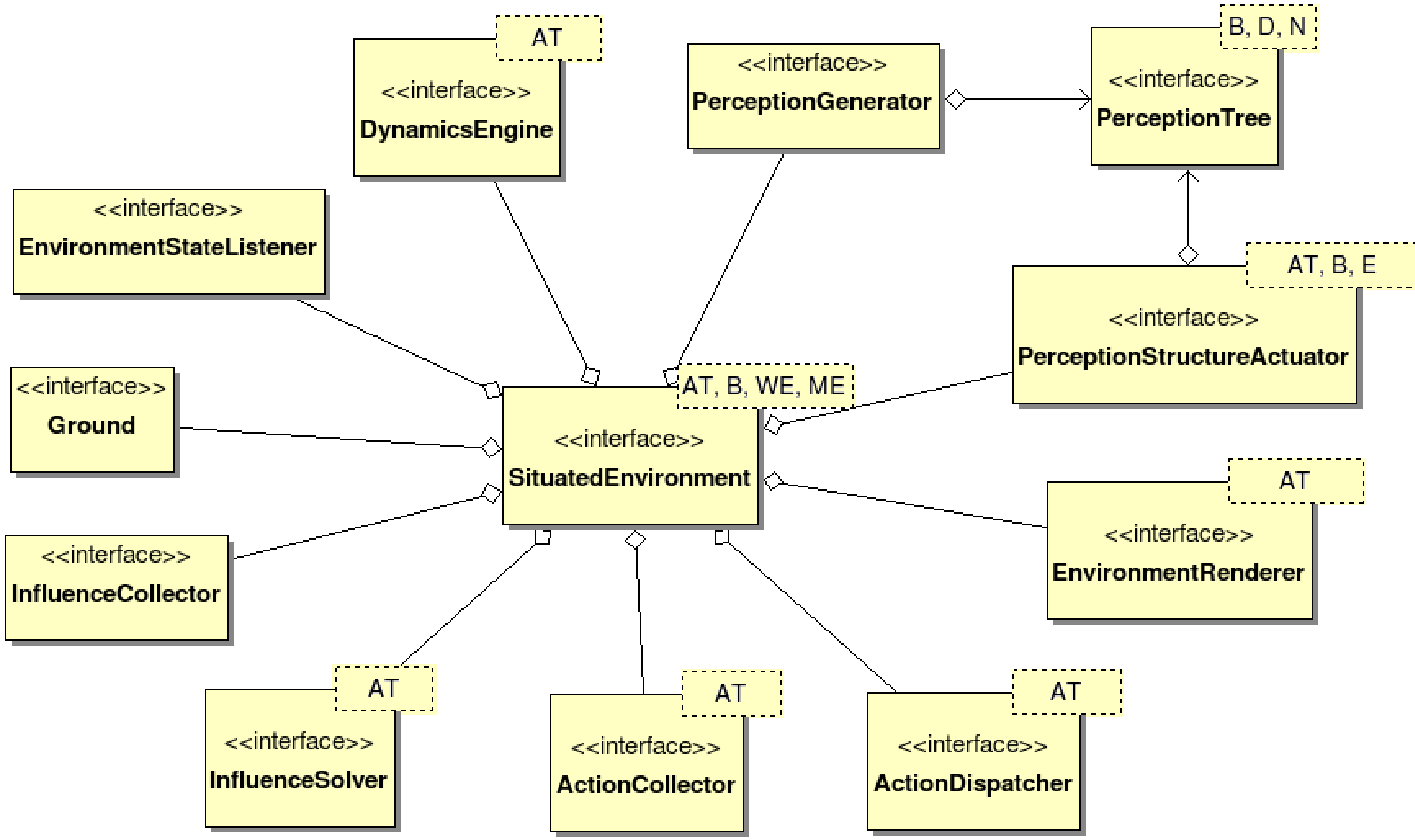
- Environment contains objects that could be:
  - Immobile/Static objects (wall...)
  - Mobile/Dynamic objects
    - managed by the environment itself
    - influenced by the characters (aka. Bodies and Avatars)
- Environment maintains a set of environmental laws (collision avoidance, gravity...)
- Environment has endogenous processes that permit to move the objects outside the scope of specific character's actions on them.

State Diagram of the Environment and the Agents

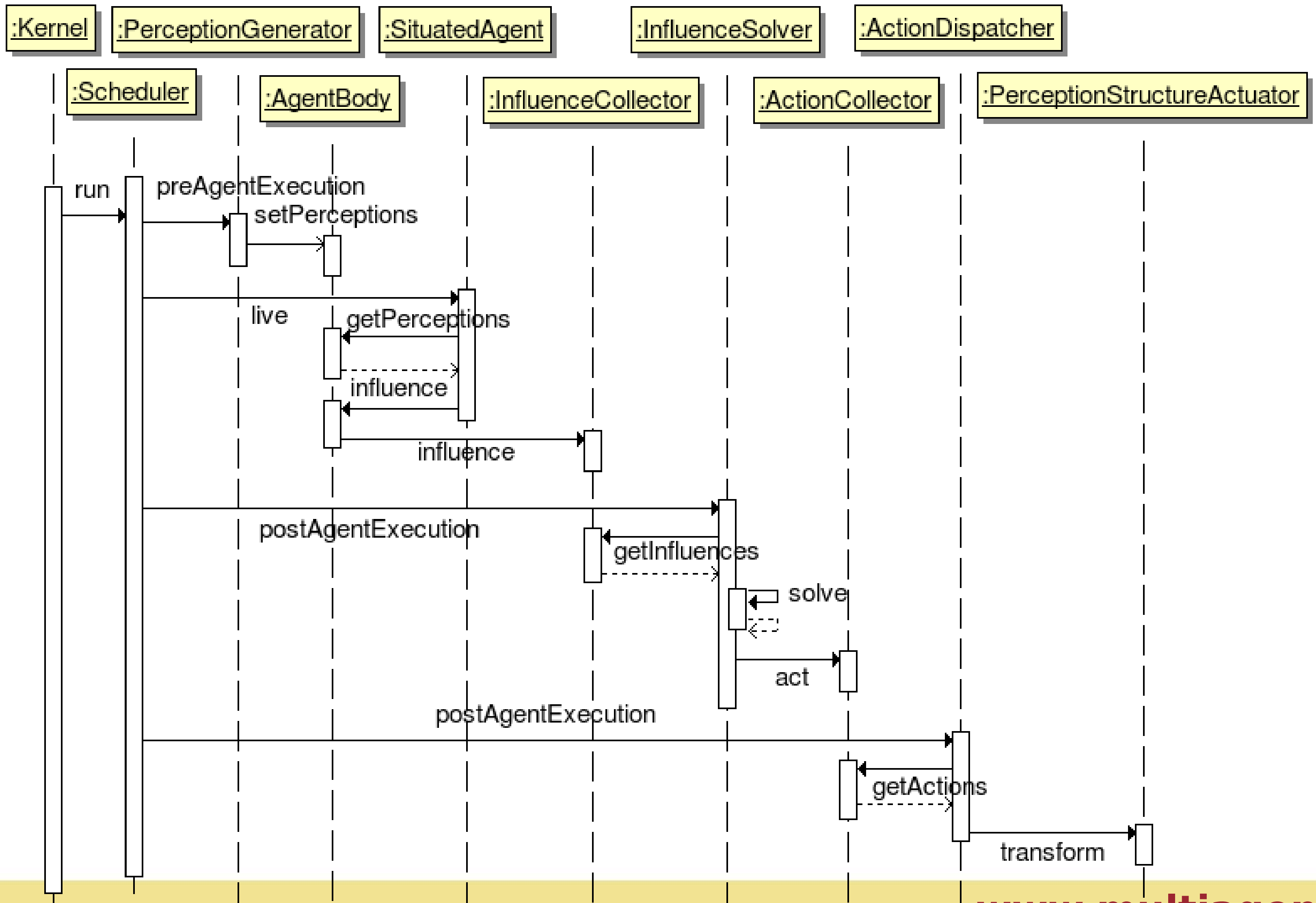




Simplified Class Diagram of the Environment Model



Sequence Diagram for One Simulation Step



Defining an Efficient Data Structure

- **It is required to define a data structure, which allows to efficiently access to the smallest set of objects concerned by a process.**
- In reality, the simulation environment will contains a set of data structures dedicated to specific processes.
- Some examples:
  - Perception of static objects
  - Perception of mobile objects
  - Ground Model: ”keep on floor” and ”traversability”
- Some classical structures are :
  - Grid (1D, 2D, 3D)
  - Tree (1D, 2D, 3D)

Grid Data Structures

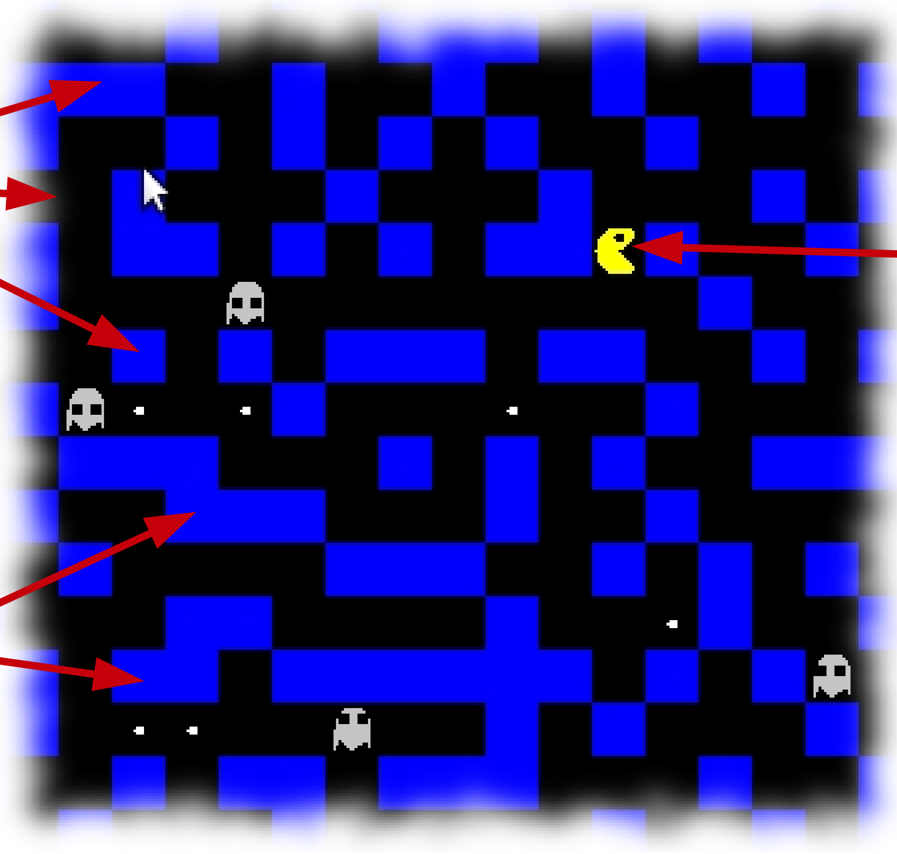
- Grid data structure (or matrix) is one of the most simple way to store environment's objects.
- It is discretizing the world into regular cells.
- Each cell contains a set of objects (basically this set is restricted to zero or one element)
- This data structure is one of the most efficient: direct access to a cell's content and its neighbours.
- But it suffers of several drawbacks:
  - limited world sizes
  - mobile objects move in a discrete way

Example of a Grid

Cell in the grid

Environmental object of type « Player »

Environmental objects of type « Obstacle »



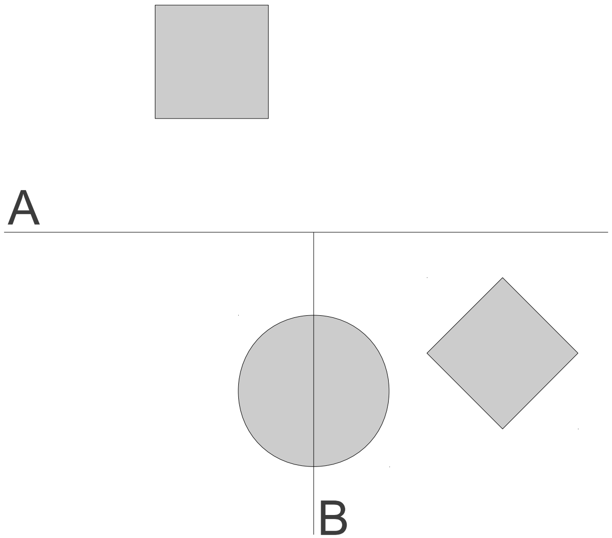
PacMan® Demo from the Janus Platform

Various Types of Trees

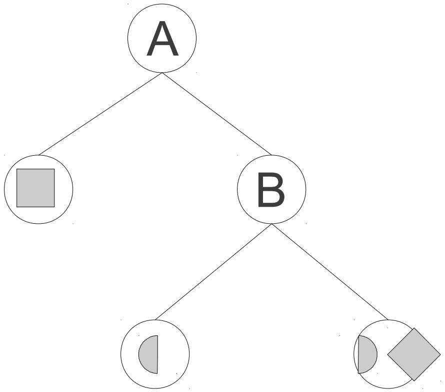
- Typical space-partitioning data structures are:
  - BSP Tree: binary space partition
    - 2-D tree (short for 2-dimensional tree): specific case of BSP trees that uses only splitting planes that are perpendicular to one of the coordinate system axes.
  - Quadtree: (4-D tree) used to partition a space by recursively subdividing it into 4 quadrants.
  - Octree: (8-D tree) used to partition a space by recursively subdividing it into 8 octants.
  - B-tree, R-tree...

Example of a 2D-Tree

- The space are divided in regions by lines.
- Each division line corresponds to a node of the tree.
- The objects are put inside the leafs.



Objects in Environment



2D-Tree of the Environment

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD www.multiagent.fr

From Multiagent Simulation to Virtual Environment Simulation

Stéphane GALLAND – stephane.galland@utbm.fr

SET Laboratoire Systèmes et Transports

utbm

version 2012.01

Basics of Movement Algorithms

- Each character has a current position and possibly additional physical properties that control its movement:
    - maximal speed, velocity...
- A movement algorithm is designed to use these properties to work out where the character should be next.

- All movement algorithms has the same basic form:
  - they take geometric and semantic data about their own state and the state of the world;
  - they come up with an action output representing the movement

Input and Output of the Motion Behavior

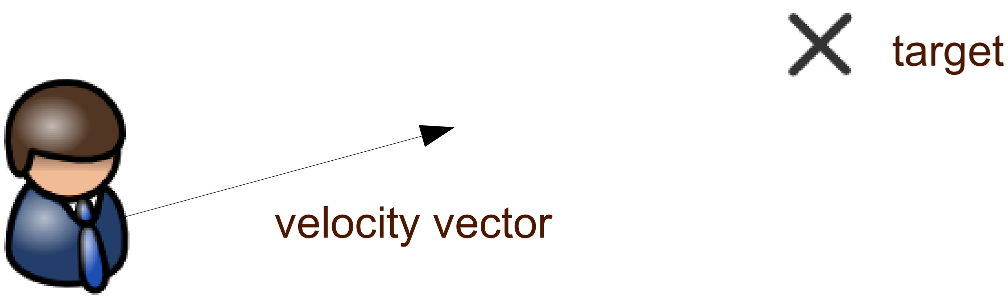
```
class AIBody
        position              # 2D or 3D vector
        orientation           # single floating point value
        linear_velocity       # 2D or 3D vector
        angular_velocity      # single floating point value
        perceptions           # list of the objects in the fov
```

Steering Behavior Algorithm

```
class SteeringBehaviourOutput
        linear_acceleration     # 2D or 3D vector
        angular_acceleration    # a single floating point value
```

Seek Algorithm Principle

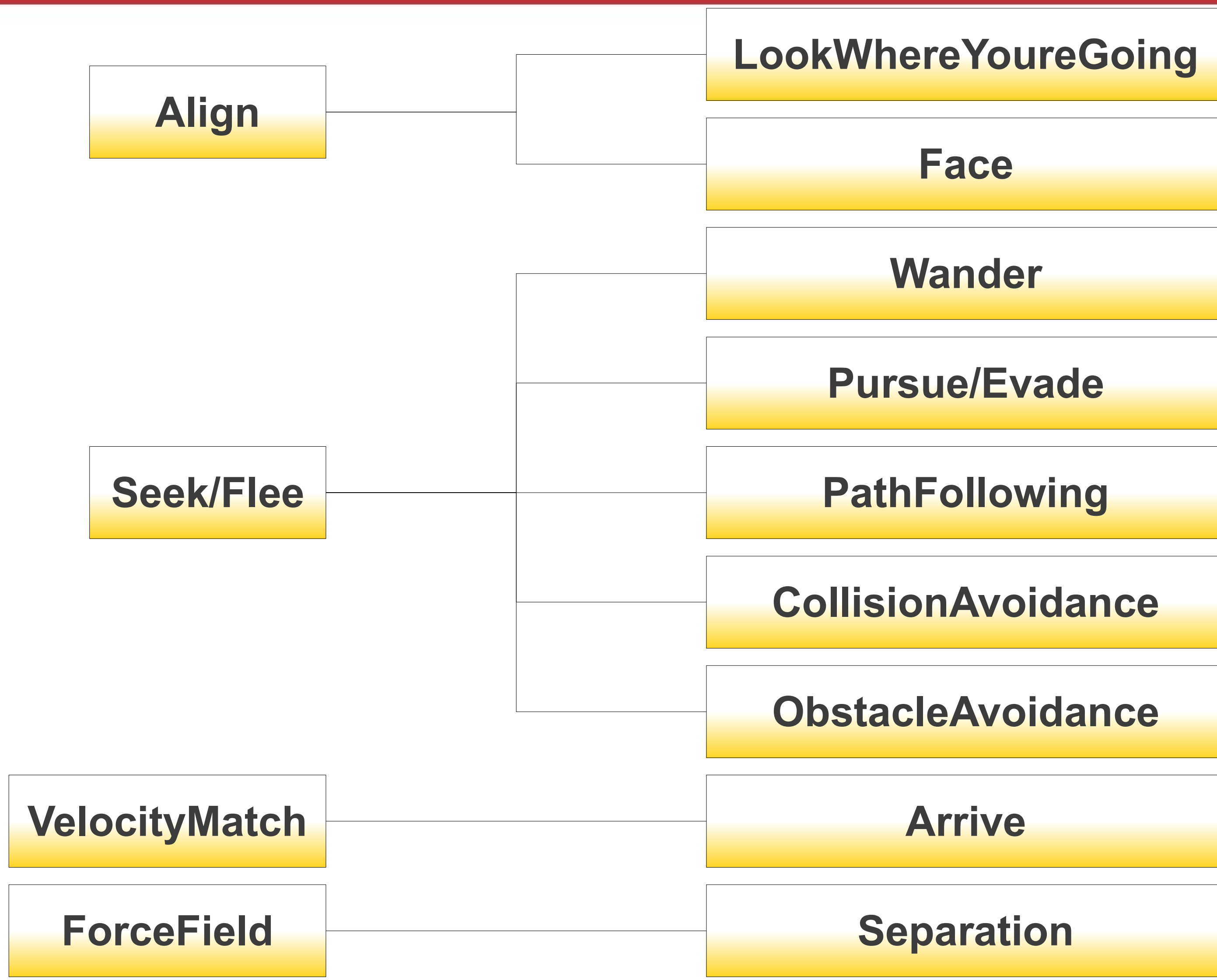
- A seek algorithm takes as input the character's attributes and target position.
- Compute the direction from the character to the target and request a velocity along this line.
- Orientation values are ignored
  - getNewOrientation function may be used



Seek Algorithm

```
class SeekBehaviour
    character       # Attribute that describes the properties of the character
    target          # the point to reach
    maxSpeed        # Holds the maximal speed the character could travel

    function run()
        # Create the structure for output
        output = new MovementOutput

        # Get the direction to the target
        output.velocity = target.position – character.position

        # The velocity is along this direction, at full speed
        output.velocity.normalize()
        output.velocity *= maxSpeed

        # Face in the direction we want to move (comment if no orientation change)
        output.orientation = atan2(-character.velocity.y, character.velocity.x)

        # Output the move data
        return output
    end function
end class
```

Summary on Steering Behaviours



Combined Steering Behaviours

- Individually steering behaviours can achieve a good degree of movement sophistication.
- But a moving character usually needs more than one steering behaviour.
- It needs to reach its goal, avoid collisions, tend toward safety as it moves, and avoid bumping into walls.
- A combination of two or more steering behaviours is named combined steering behaviour.
- Two major approaches for combining behaviours:
  - blending: use priorities or weights to select a behaviour to run
  - arbitration: run all the behaviours and select the best result

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

www.multiagent.fr

From Multiagent Simulation to Virtual Environment Simulation

Stéphane GALLAND – stephane.galland@utbm.fr

SET Laboratoire Systèmes et Transports

utbm

version 2012.01

Demos

Simulation of a Metro station
© 2009, Voxelia

Demos

Simulation of the place at the front of the Belfort's rail station
© 2011, Voxelia

Demos

Simulation of the Isaac Rabin Place at Belfort
© 2012, Voxelia

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

www.multiagent.fr

Janus and Jaak Platforms

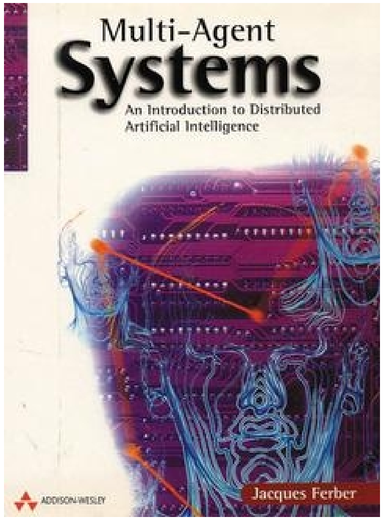
Stéphane GALLAND – stephane.galland@utbm.fr

Appendix <

SET
Laboratoire Systèmes et Transports

utbm

Bibliography



Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence

Jacques FERBER

1999

Addison Wesley, London

ISBN 0-20-1360489

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

Bibliography

Artificial Intelligence for Games

Ian Millington

2006

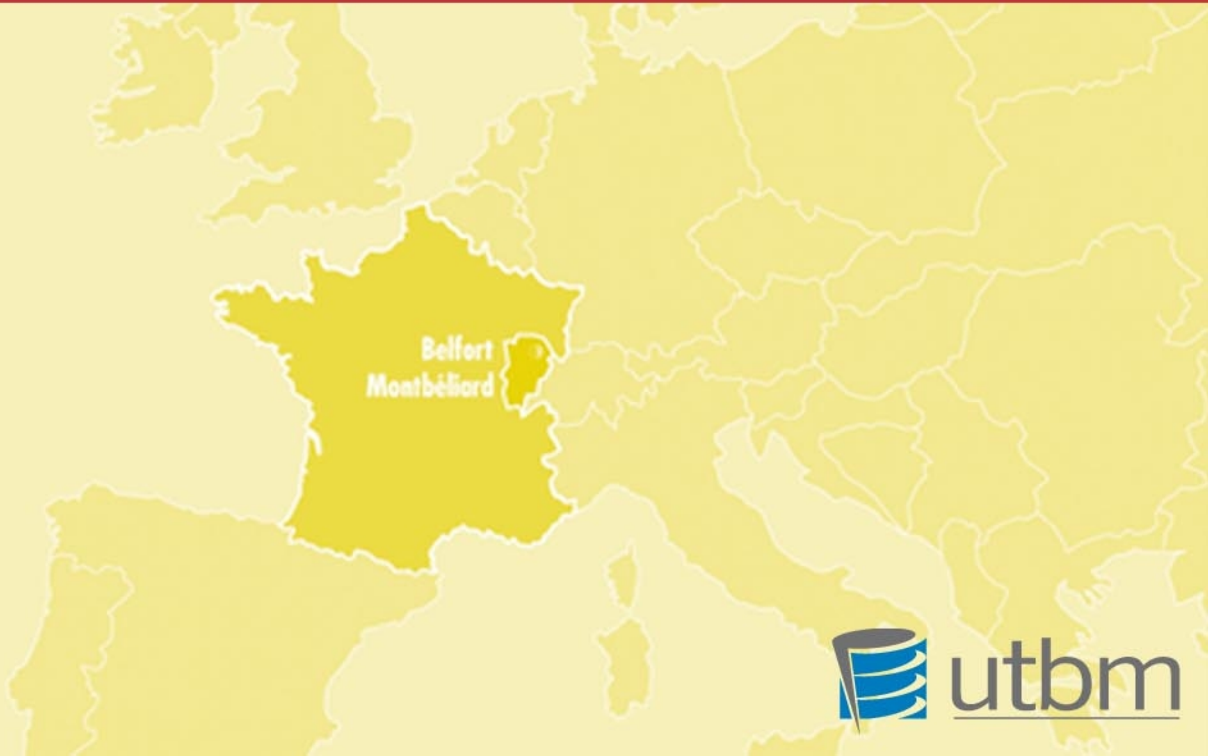
Morgan Kaufmann Publishers
Elsevier Science

ISBN 0-12-497782-0

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

Stéphane GALLAND
http://www.multiagent.fr/People:Galland_stephane
stephane.galland@utbm.fr

www.utbm.fr



SET
Laboratoire Systèmes et Transports

utbm