IIF
INSTITUT FÜR
INFORMATIK
FREIBURG

Practical / Laboratory (SS12)

Multi-Agent Programming

Dr. C. Becker-Asano, Dr. S. Wölfl & Prof. Dr. B. Nebel
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Foundation of Artificial Intelligence

UNI
FREIBURG

IIF
INSTITUT FÜR
INFORMATIK
FREIBURG

Overview

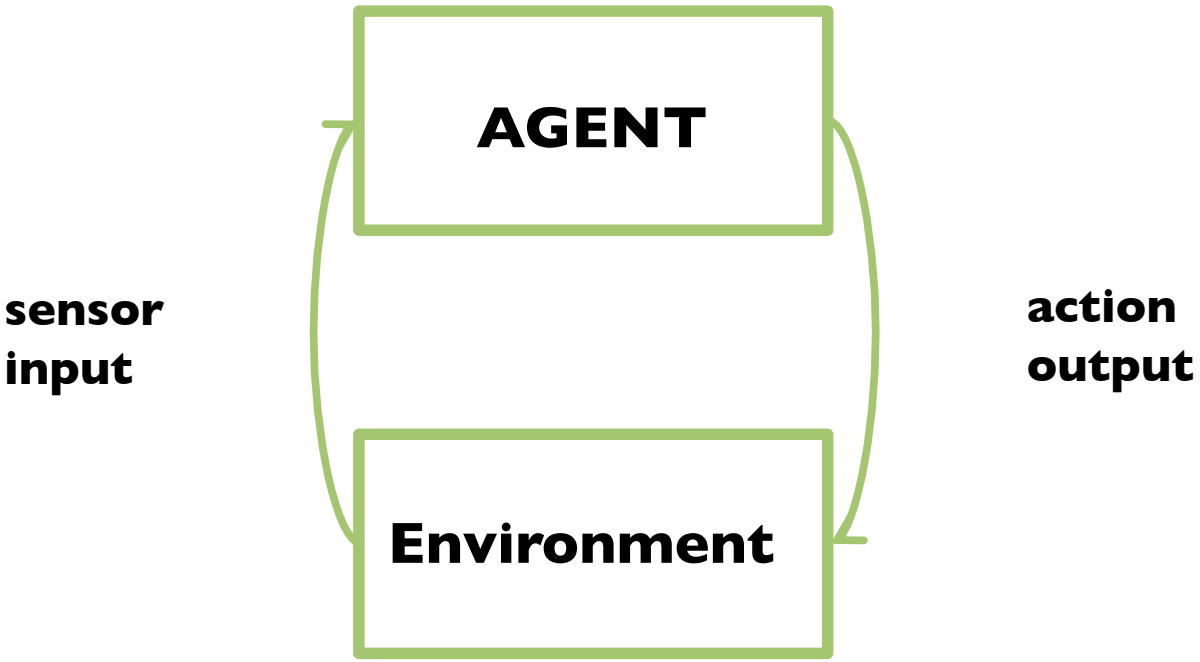
- What is an agent?
- What is a multi-agent system?
- What are applications for agents and multi-agent systems?
- Practical realization of theoretical concepts

Slides content and figures are taken from
Michael Wooldridge, *Introduction to MultiAgent Systems*,
John Wiley and Sons Ltd, February 2002

UNI
FREIBURG

What is an agent?

An agent is a computer system that is **situated** in some environment, in which it is capable of **autonomous** action, in order to meet its predefined **objectives**. (Wooldridge & Jennings, 1995)



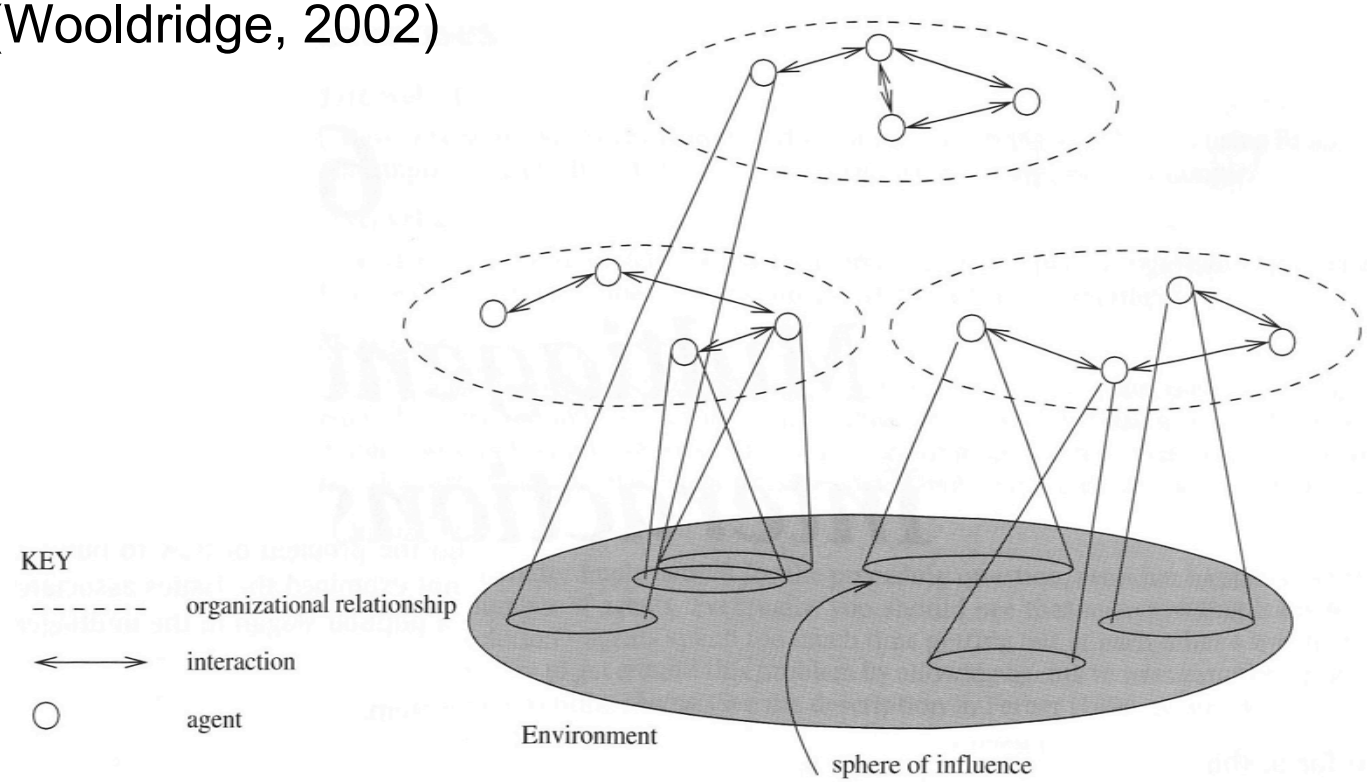
What is an agent?

Intelligent Agents: (Wooldridge & Jennings, 1995)

- **Reactivity:** Intelligent agents are able to perceive their environment, and respond in timely fashion to changes that occur in it in order to satisfy predefined objectives.
- **Proactiveness:** Intelligent agents are able to exhibit goal-directed behavior by taking the initiative in order to satisfy predefined objectives.
- **Social ability:** Intelligent agents are capable of interacting with other agents (and possibly humans) in order to satisfy predefined objectives.

What is a multi-agent system?

A multiagent system consists of a number of agents, which interact with one another. In order to successfully interact, these agents will require the ability to cooperate, coordinate, and negotiate with each other, in much the same way that we do. (Wooldridge, 2002)



(Wooldridge 2002, p.106)

Example application (1)

Multi-agent information retrieval system

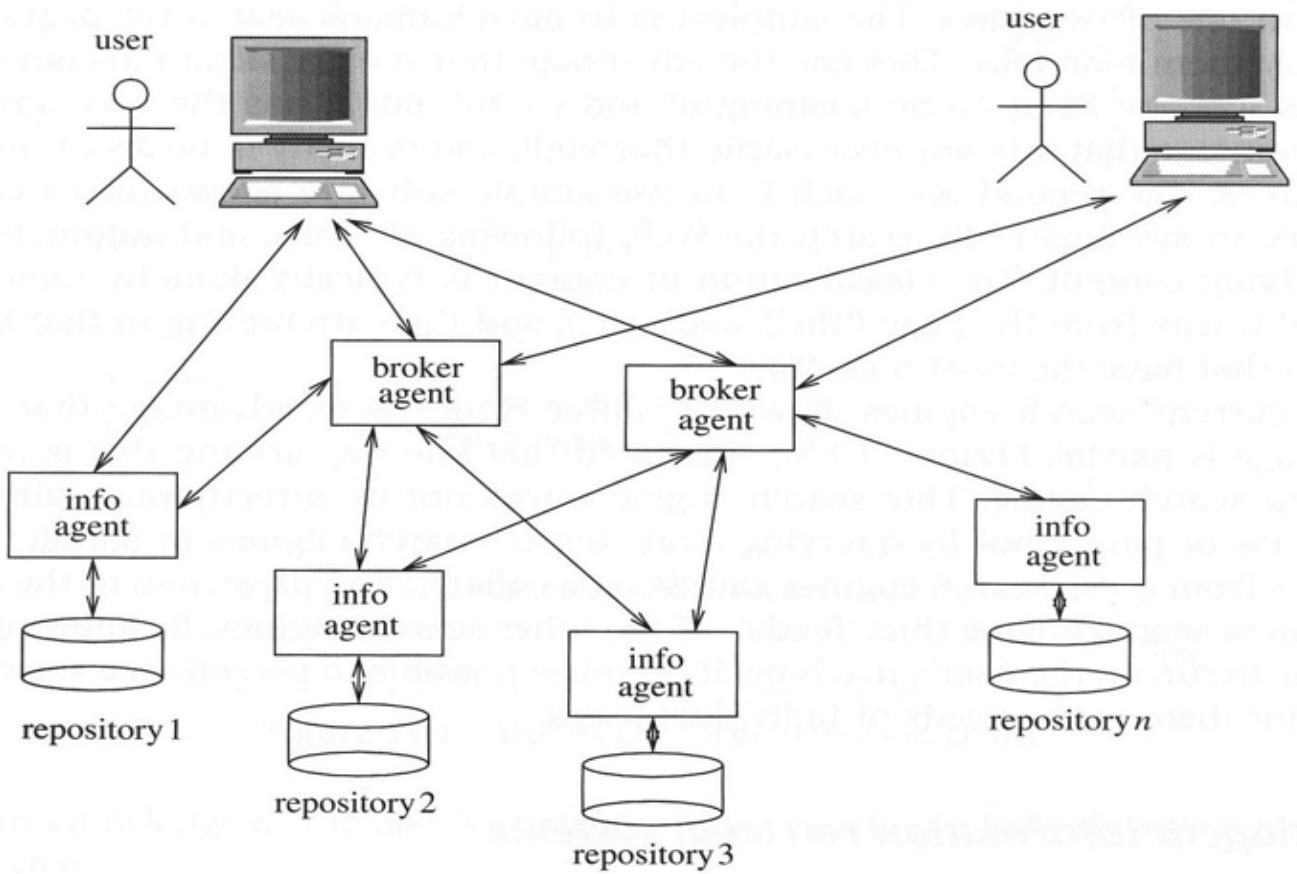
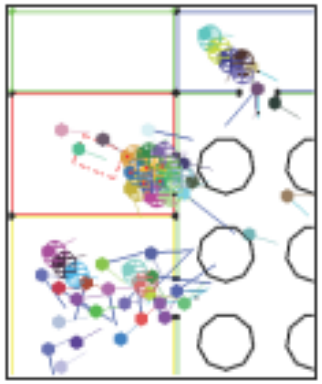


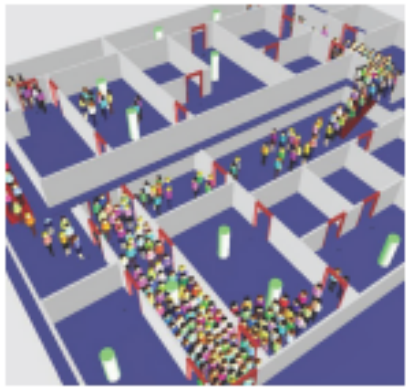
Figure 11.2 Typical architecture of a multiagent information system.

(Wooldridge 2002, p. 254)

Example application (2)

- Agents for Social Simulation
- Properties that are difficult to observe in nature may be studied at leisure in isolation, recorded, and then 'replayed' if necessary.
- Related to the field called „Socionics“
- E.g. „crowd simulation“





(Pelechano, N., Allbeck, J.M., Badler, N.I., 2007)

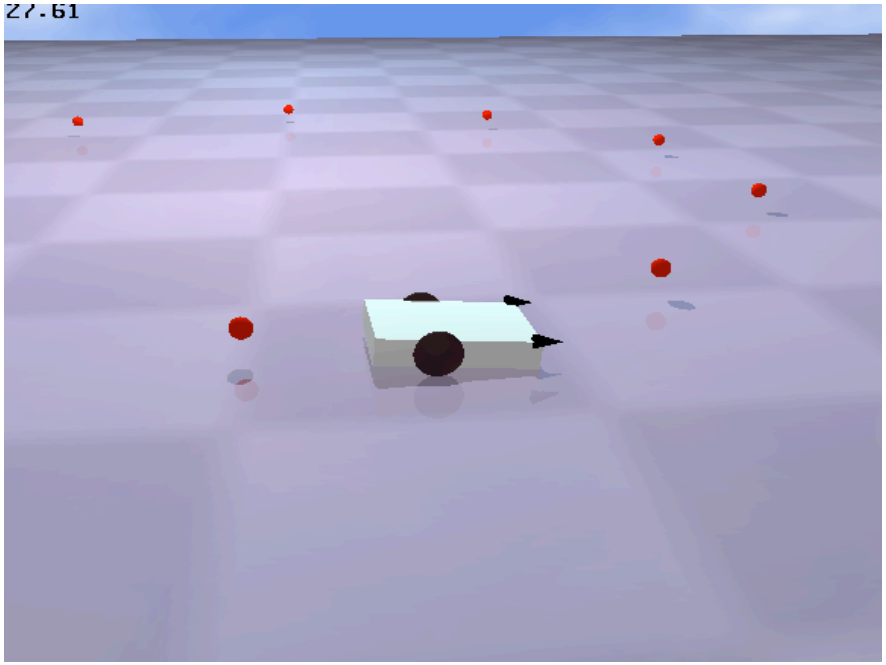
Practical realization of theoretical concepts

- Multi-agent software „breve“
  - Provides a complete development environment
  - Is open source
  - Runs in (Mac,) Linux, MS Windows environments
  - Runs in 3D (including physics simulation, if needed)
- Programming language „Steve“
  - Object-oriented scripting language
  - interpreter language

IIF
INSTITUT FÜR
INFORMATIK
FREIBURG

First Scenario: Braitenberg vehicles in breve

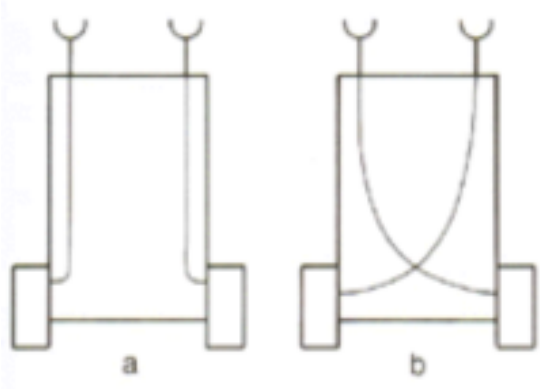
- What is a Braitenberg vehicle?
- How to simulate a Braitenberg vehicle in breve?
- How is breve structured?



UNI FREIBURG

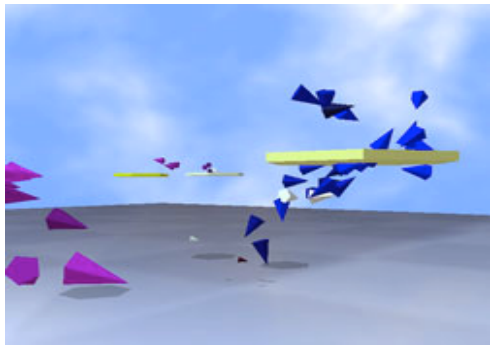
What is a Braitenberg Vehicle?

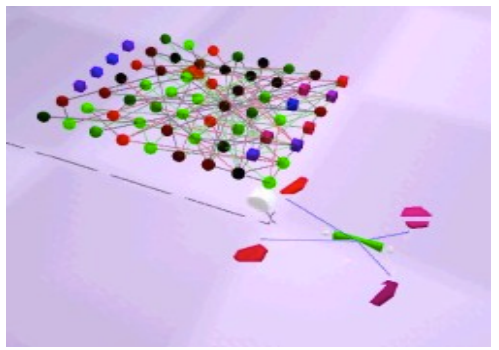
- small robots exhibiting complex behaviors with very simple circuitry
- typically small box-shaped bodies with one wheel on each side and sensors detecting different types of stimuli (e.g., light)
- sensors connected directly to wheels, so that they turn when sensors are activated
- vehicles can appear to exhibit behaviors such as “love” or “hate”



Introduction to breve

- Breve (http://www.spiderland.org/)
- free, open-source software project for 3D visualization based on OpenGL
- available for Linux, Windows, and Mac OS X
- object-oriented programming in 'Steve‘
- For the sources, see also: https://github.com/kephale/breve/





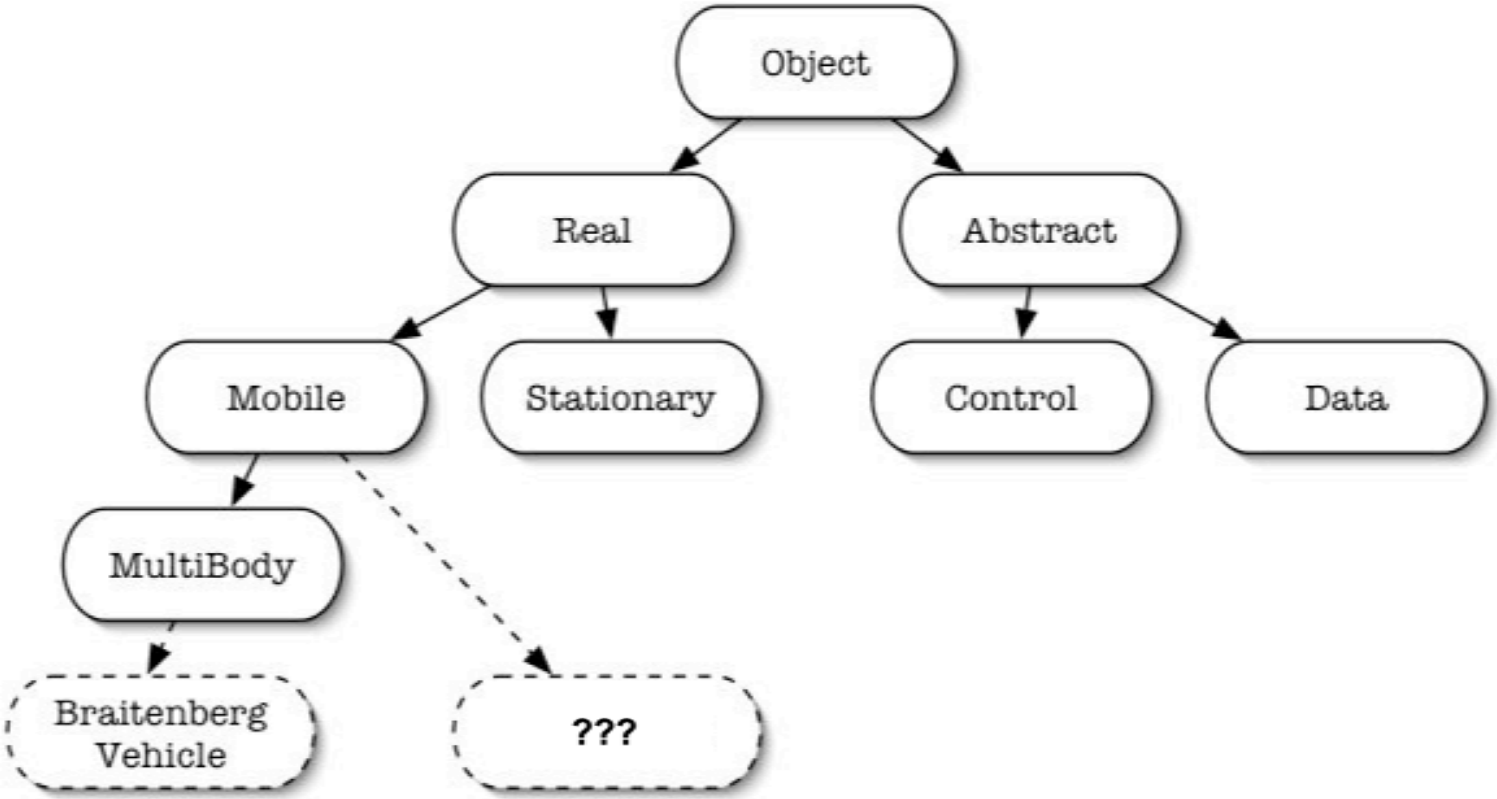
IIF
INSTITUT FÜR
INFORMATIK
FREIBURG

Breve: Object-oriented programming

- **Objects** consist of **data** and **methods**
- **Class** defines the specific **type** of an object
- Concrete instances of classes form agents and objects in the world
- Classes inherit **behaviors** (methods) of their parent class
  - Implementing Agents by inheriting existing breve-classes saves us work
- Agent is an instance of the parent class **Real**
- Instances of parent class **Abstract** do not appear in the world

UNI FREIBURG

Class hierarchy



Programming in Steve

1. What is a Controller Object?
2. How to define classes?
3. How to define methods?

1. What is a controller object?

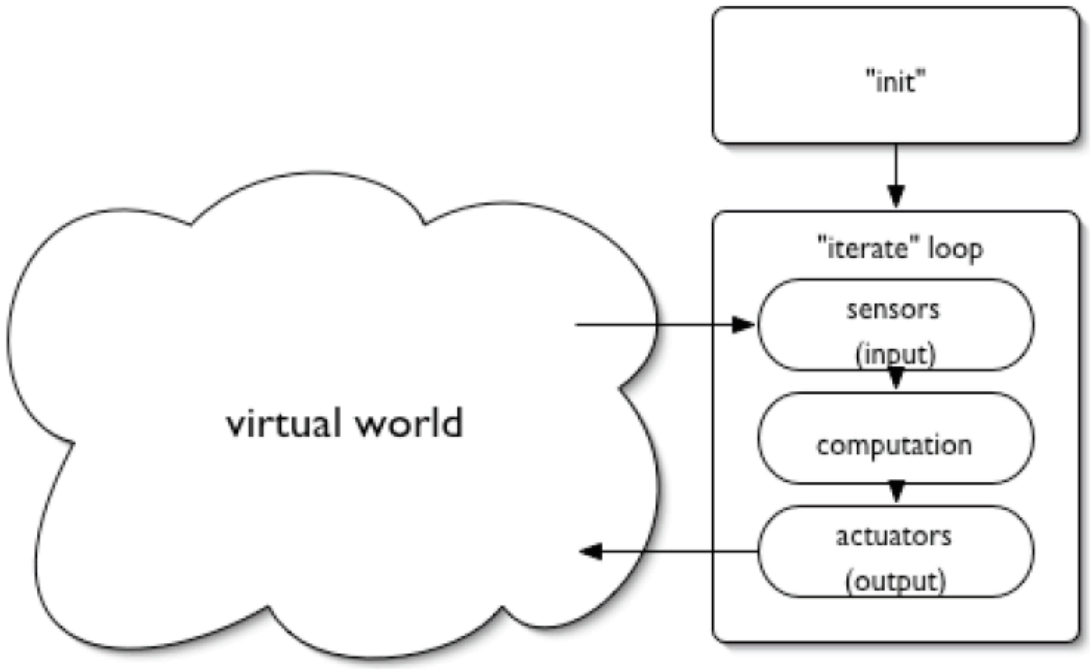
- Similar to a main - method
- Needs to be sub-class of class Control
- Is automatically created at the beginning of simulation
- Provides agents with services
- Coordinates communication between agents and simulation software

```
Controller myBraitenbergControl.
BraitenbergControl : myBraitenbergControl
{
 + variables:
            vehicle (object).
            (...)
 + to init:
            vehicle = new BraitenbergVehicle.
            (...)
}
```

2a. How to define classes

Methods & Data structures:

```
ParentClass : ClassName {
        + varia
                (
        + to in
                (
        + to (
                (
(for agents:)
        + to in
                (
}
```



2b. Create a new instance

- init: is executed at each instantiation
- iterate: is executed at each iteration step in simulation
- instances are created by defining an object definition:
  - new Instance-Name.
  - Examples:
    - new BraitenbergVehicle.
    - 10 new BraitenbergLights.

3. How to define methods?

- Method without parameters

```
+ to MethodName:
```

- Method with parameters - for each parameter define the following triple:

1. Keyword 2. Variablename 3. Type

```
+ to set-rot of-joint joint (Object) to-value value (float):
```

- Calling a method:

```
myObject set-rot of-joint myJoint to-value 200.
myObject set-rot to-value 200 of-joint myJoint.
```

- Meaningful denotations increase your source code's readability and reusability
- See: http://www.clean-code-developer.com/AllPages.aspx

Breve: Summary

- We implement agent classes and specify the behaviors by using init, iterate, and own methods.
- We create a 'Controller' object and instantiate objects within the init part by using the keyword new.
- More structures:
  - self - reference to the current object
  - controller - reference to Controller object
  - super - reference to parent (super) class

IIF
INSTITUT FÜR
INFORMATIK
FREIBURG

References

- N. R. Jennings and M. Wooldridge (1995).
  Applying Agent Technology.
  In: *Journal of Applied Artificial Intelligence.*
  Special issue on Intelligent Agents and Multi-Agent Systems.
- M. Wooldridge (2002).
  An Introduction to Multiagent Systems.
  John Wiley and Sons Ltd, February 2002.
- J. Klein (2002).
  BREVE: a 3D Environment for the Simulation of Decentralized Systems and Artificial Life.
  Proceedings of Artificial Life VIII.
- N. Pelechano, J. M. Allbeck, N. I. Badler (2007).
  Controlling individual agents in high-density crowd simulation.
  Proceedings SCA ‘07

UNI FREIBURG

Mars Explorer World

- look into „folders2copy\MarsExplorerScenario“