Sebastian Rodriguez
(On behalf of SARL Team)

Semana de la Ingeniería 2014
Facultad Regional Tucumán
Universidad Tecnológica Nacional
June 6th, 2014

sebastian.rodriguez@gitia.org
Agent Technology - (really!) Brief Overview

1. Trends in computer science
2. Agents

SARL Team

2. SARL Team

Design Principles

3. Design Principles

Programming MAS in SARL

4. Definitions
5. Built-in Capacities
6. Environment

Summary & Future works

5. Summary & Future works
History of Computing

Five ongoing trends have marked the history of computing

- Ubiquity;
- Interconnection;
- Intelligence;
- Delegation;
- Human-orientation: easy/natural to design/implement/use.

Other Trends in Computer Science

- Grid Computing;
- Ubiquitous Computing;
- Semantic Web.
Programming has progressed through:

- machine code;
- assembly language;
- machine-independent programming languages;
- sub-routines;
- procedures & functions;
- abstract data types;
- objects;

to

agents.
Agent: a new paradigm?

- Agent-Oriented Programming (AOP) reuses concepts and language artifacts from OOP.
- It also provides an higher-level abstraction than OOP.
Multiagent systems: a new view? Which characteristics?

- Multiagent-based approach (metaphor or paradigm) is represents a new way of analyzing, designing and implementing software systems, especially complex systems.
- It strongly improves/impacts the way in which people conceptualizes and implements a large number of systems.
- Strong interdisciplinary inspiration: social and biological sciences, Economics and Game theory, control theory.
- Large panel of application.
Agent: a first Definition

No commonly/universally accepted definition.

Agent [Wooldridge and Ciancarini, 2001]

An agent is an entity with (at least) the following attributes / characteristics:

- Autonomy
- Reactivity
- Pro-activity
- Social Skills - Sociability
Agents and Environment

Agent and Environment

- located in an environment (situatedness)
- perceives the environment through its sensors.
- acts upon that environment through its effectors.
- to maximize progress towards its goals.
Outline

1. Agent Technology - (really!) Brief Overview
   - Trends in computer science
   - Agents

2. SARL Team

3. Design Principles

4. Programming MAS in SARL
   - Definitions
   - Built-in Capacities
   - Environment

5. Summary & Future works
A whole team

Design & Development

- Nicolas GAUD
- Stéphane GALLAND
- Sebastian RODRIGUEZ

Contributions and Ideas (a lot !!)

- Olivier BOISSIER
- Vincent HILAIRE
- Flavien BALBO
- Gauthier PICARD
- Luk KNAPPEN
- Jomi HÜBNER

Many others, every contributor is welcome 😊
Contributions and uses during the Young Age

Agent Technology - (really!) Brief Overview
Trends in computer science
Agents

SARL Team
Design Principles

Programming MAS in SARL
Definitions
Built-in Capacities
Environment

Summary & Future works

Energy mananagement & simulation

Energy, Transport & LUTI simulation

Environment definition & transport simulation

Transport management & simulation
1. Agent Technology - (really!) Brief Overview
   - Trends in computer science
   - Agents

2. SARL Team

3. Design Principles

4. Programming MAS in SARL
   - Definitions
   - Built-in Capacities
   - Environment

5. Summary & Future works
Design Principles

- Clear separation between Language and Platform related aspects.
- Everything is distributed and it should be transparent.
- Massively parallel.
- Event-driven interactions.
- All agents are holonic.
- Platform- and architecture-independent.
- There is not only one way of interacting but infinite.
- Coding should be fun (Ruby/Scala-like 🌻).
Expectations for SARL

- Stop implementing Agents with Object-Oriented concepts.
- Playground to find minimal Agent-Oriented Programming concepts.
- Agents should be simple to extend.
- Provide the community a common discussion forum. (a testbed)
1. Agent Technology - (really!) Brief Overview
   - Trends in computer science
   - Agents

2. SARL Team

3. Design Principles

4. Programming MAS in SARL
   - Definitions
   - Built-in Capacities
   - Environment

5. Summary & Future works
### Overview of SARL Concepts

**a MAS in SARL**

A collection of Agents interacting together in a collection of shared distributed Spaces.

#### 4 main concepts
- Agent
- Capacity
- Skill
- Space

#### 3 main dimensions

| Individual: the Agent abstraction (Agent, Capacity, Skill) |
| Collective: the Interaction abstraction (Space, Event, etc.) |
| Hierarchical: the Holon abstraction (Context) |

**SARL: a general-purpose agent-oriented programming language.**

Agent

- An agent is an autonomous entity having some intrinsic skills to implement the capacities it exhibits.
- An agent defines a Context.
- An agent initially owns native capacities called Built-in Capacities.

```java
/* Package & Import Section Omitted */
agent HelloAgent{
    uses Lifecycle, Schedules

    on Initialize {
        System.out.println("Hello World!")
    } in(2000) [killMe]

    on Destroy {
        System.out.println("Goodbye World!")
    }
}
```
Capacities and Skill

Capacity


Action

- A specification of a transformation of a part of the designed system or its environment.
- Guarantees resulting properties if the system before the transformation satisfies a set of constraints.
- Defined in terms of pre- and post-conditions.

Skill

A possible implementation of a capacity fulfilling all the constraints of its specification.
Capacities and Skill

capacity Logging {
    def debug(s: String)
    def info(s: String)
}

skill BasicConsoleLogging implements Logging {
    def debug(s: String) {
        System.out.println("DEBUG:"+s)
    }
    def info(s: String) {
        System.out.println("INFO:"+s)
    }
}

agent HelloAgent{
    uses Lifecycle, Schedules, Logging
    on Initialize {
        setSkill(Logging,new BasicConsoleLogging());
        info("Hello World!");
        in(2000)[killMe]
    }
    on Destroy {
        info("Goodbye World!")
    }
}
Interactions between Agents

Space
Support of interaction between agents respecting the rules defined in various Space Specifications.

Space Specification
- Defines the rules (including action and perception) for interacting within a given set of Spaces respecting this specification.
- Defines the way agents are addressed and perceived by other agents in the same space.
- A way for implementing new interaction means.
Default Space: an Event Space

- Event-driven interaction space.
- Default Space of a context, contains all agents of the considered context.
- Event: the specification of some occurrence in a Space that may potentially trigger effects by a participant.
Ping - Pong
Exchanging information between agents

/* Events Definition */

```java
event Ping {
    var value : Integer
    new (v : Integer) {
        value = v
    }
}
event Pong {
    var value : Integer
    new (v : Integer) {
        value = v
    }
}

/* Receive a ping, send a pong. */
agent PongAgent {
    uses Lifecycle, DefaultContextInteractions
    on Initialize {
        println("Waiting for ping")
    }
    on Ping {
        println("Recv Ping: "+occurrence.value)
        println("Send Pong: "+occurrence.value)
        emit(new Pong(occurrence.value))
    }
}
```

/* Send a ping. */

```java
agent PingAgent{
    uses Lifecycle, Schedules,
    DefaultContextInteractions

    var count : Integer

    on Initialize {
        println("Starting PingAgent...")
        count = 0
        in(2000) [sendPing]
    }
    def sendPing {
        if (defaultSpace.participants.size >1) {
            emit(new Ping(count))
            count = count + 1
        } else {
            in(2000) [sendPing]
        }
    }
    on Pong {
        in(1000) [
            println("Send Ping: "+count)
            emit(new Ping(count))
            count = count + 1
        ]
    }
}
```
Context and Interactions

- Defines the boundary of a sub-system.
- Collection of Spaces.
- Every Context has a Default Space.
- Every Agent has a Default Context, the context where it was spawned.
Spaces and Contexts

Contexts and Holonic properties

- All agents have at least one External Context (the default one).
- All agents participate in the Default Space of all Contexts they belong to.
- The Janus Context is omnipresent.
A SRL Agent has inherently a set of **Built-in Capacities**

### Current Built-in Capacities

- **ExternalContextAccess**
- **InnerContextAccess**
- **Behaviors**
- **Lifecycle**
- **Schedules**
- **DefaultContextInteractions**
Dimensions of the Environment

- Execution
- Physic
- Social

Key Ideas

- It is omnipresent
- Agents can interact with it via Capacities and Spaces
- Manages access to resources and structures
### Execution Environment requirements

- Handles Agent’s Lifecycle
- Provides Built-in Capacities
- Implements SARL concepts
- Handles resources

### Janus as SARL Execution Environment

- Fully distributed.
- Dynamic discovery of Kernels.
- Automatic synchronization of kernels’ data (easy recovery).
- Micro-Kernel implementation.
Physical Dimension

Physical Environment

Class of real or simulated systems in which agents and objects have an explicit position, and that produce localized actions.

Properties

- Contains all objects
- Agents interact with it via dedicated Capacities
- Agents’ Bodies are “managed” by the Environment
- Multiple “Views” of the environment can be implemented (1D, 2D, 3D)
- Enforces Universal Laws (e.g. Laws of physics)
Social Dimension

- Multiple ways of agent interaction
- Supported by Space / SpaceSpecification
- Default Interaction Space: based on events (may be redefined).
- Programmer can create new SpaceSpecifications (and ways of interacting):
  - FIPA
  - Organizational (MOISE, CRIO, etc)
- Social Dimension may influence other dimensions
Integration of the Environment Dimensions

- Enables “communication” between dimensions
  - Constrains of actions/interactions from other dimensions
  - Different perceptions of the same Event according to the dimension
  - Interactions / Perceptions due to interactions in other dimensions
- Seen as Monolithic by Agent accessed by dedicated Capacities and Spaces (Holonic view)
- Should provide a uniform interface of the environment (Simulation vs Real World)
1. Agent Technology - (really!) Brief Overview
   - Trends in computer science
   - Agents

2. SARL Team

3. Design Principles

4. Programming MAS in SARL
   - Definitions
   - Built-in Capacities
   - Environment

5. Summary & Future works
What does SARL currently provide?

- Defines general agent-technology concepts: Context, Space, Agent, Capacity, Skill.
- All agents are holons.
- Intuitive Syntax (and the associated Eclipse-based IDE).
- Extensible (Capacities and Skill)
- No single way of interacting imposed.
- Janus as SARL platform
Agent Architectures:

- Capacities for Reasoning Agent: knowledge, plans, goals definition, multiple “reasoning engines”, etc.

Environments:

- Define the organizational extensions firstly based on CRIO then MOISE 😊: static and dynamic (normative).
- Continue work on physical environment integration, and its interaction with the other dimensions.

Language:

- Enforcing Pre- and post-conditions.
- Formal specification.
- Define grammar extensions for describing the environments instances.
Join Us

The whole is greater than the sum of its parts

Join Us

- Open Source Project - Apache v2 License
- SARL
  - http://www.sarl.io
  - http://www.github.com/sarl
- Janus Project
  - http://www.janusproject.io
  - http://www.github.com/janus-project
- Every contributor is welcome 😊
Distributed Artificial Intelligence at FRT-UTN
Join us here at FRT-UTN

**GITIA**

- UTN Research Group
- International Collaborations
- Grad and Undergrad projects
- Agent oriented Software Engineering

**DAI Courses**

- Undergrad (Second Semester)
- Masters
- Ph.D.

Agent Technology - (really!) Brief Overview
Trends in computer science
Agents

SARL Team

Design Principles

Programming
MAS in SARL
Definitions
Built-in Capacities
Environment

Summary & Future works

S. Rodriguez
SARL: Agent Programming Language
SARL
Agent-Oriented Programming Language
www.sarl.io

Sebastian Rodriguez
(On behalf of SARL Team)

Semana de la Ingeniería 2014
Facultad Regional Tucumán
Universidad Tecnológica Nacional
June 6th, 2014

sebastian.rodriguez@gitia.org
6 Bibliography
SARL: a general-purpose agent-oriented programming language.
Warsaw, Poland. IEEE Computer Society Press.

Agent-oriented software engineering: The state of the art.