From Ontology to Action: Streamlining Multiagent System Development with SPADE

AI2Future

Bogdan Okreša Đurić

University of Zagreb Faculty of Organization and Informatics Artificial Intelligence Laboratory

18 October 2024

Contents

Introduction

Introduction

Assistant Professor at the University of Zagreb Faculty of Organization and Informatics, and a member of the Artificial Intelligence Laboratory at UNIZG FOI. Main scientific interests can be found in:

- multiagent systems,
- semantic modelling,
- gamification,
- artificial intelligence,
- computer games.

Introduction

One of the teachers of the following courses in Croatian or English:

- Multiagent Systems,
- Database Theory,
- Declarative Programming,
- Introduction to Artificial Intelligence,
- Introduction to Computer Games,
- Internet Security,
- Computer Game Development Platforms.

Introduction

Engaged in international activities and promoting international relations:

- Erasmus student at Karl-Franzens University of Graz (AT),
- Erasmus intern at Jožef Stefan Institute in Ljubljana (SI),
- Erasmus+ intern at Elettra Sincrotrone in Trieste (IT),
- 3-month research stay at Universitat Politècnica de València in Valencia (ES),
- ITEC student at Centre for Development of Advanced Computing in NOIDA (IN),
- 16-month research visit at Universitat Politècnica de València in Valencia (ES),

Introduction

Publications

Project activity

BSc and MA Mentees

Artificial Agents

Agent

Figure 1: Visual definition of an artificial agent, based on [1, p. 55]

Artificial Agents

Figure 1: Visual definition of an artificial agent, based on [1, p. 55]

Artificial Agents

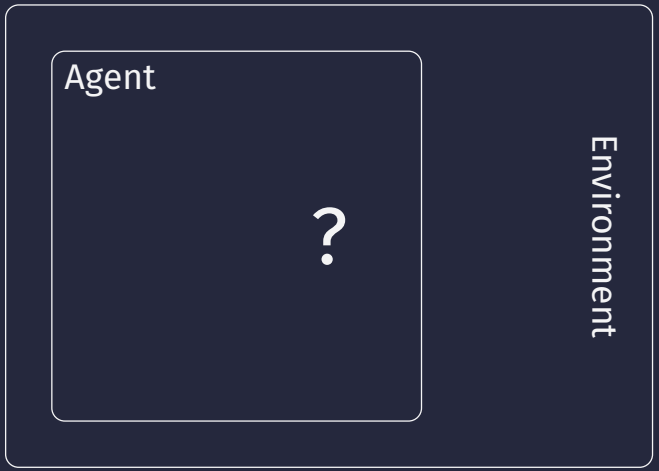

Figure 1: Visual definition of an artificial agent, based on [1, p. 55]

Artificial Agents

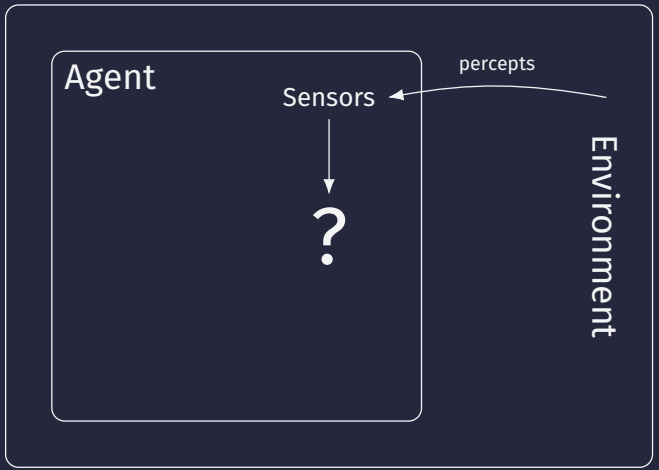

Figure 1: Visual definition of an artificial agent, based on [1, p. 55]

Artificial Agents

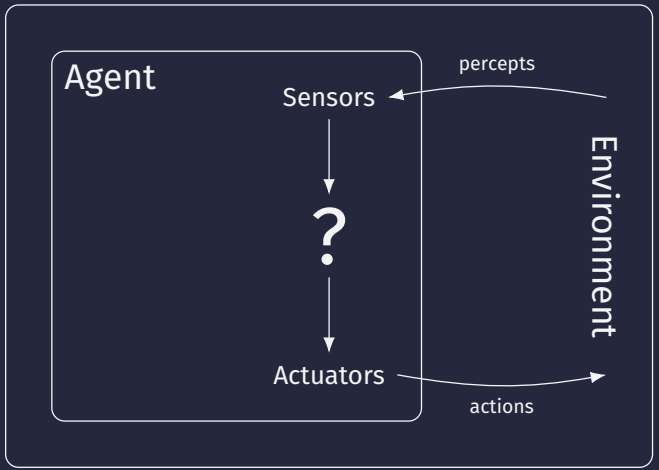

Figure 1: Visual definition of an artificial agent, based on [1, p. 55]

Artificial Agents

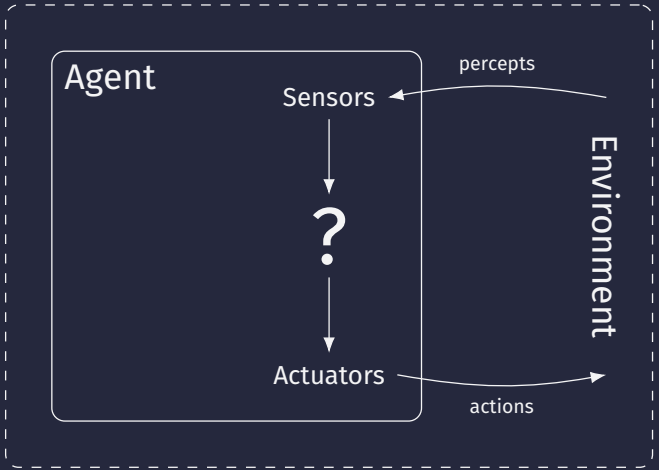

Figure 1: Visual definition of an artificial agent, based on [1, p. 55]

SPADE

Smart Python Agent Development Environment (SPADE)

```python
import spade

class DummyAgent(spade.agent.Agent):
    async def setup(self):
        print("Hello World! I'm agent {}".format(str(self.jid)))

async def main():
    dummy = DummyAgent("your_jid@your_xmpp_server", "your_password")
    await dummy.start()

if __name__ == "__main__":
    spade.run(main())
```

Listing 1: A simple SPADE agent

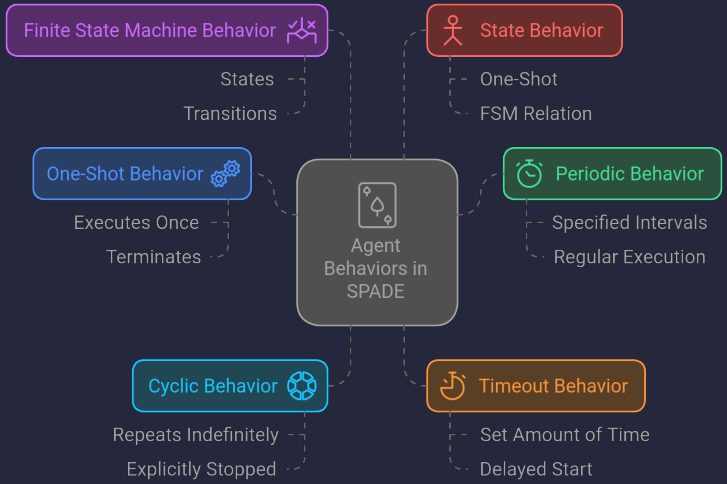

Figure 2: Types of agent behaviour in SPADE

Smart Python Agent Development Environment (SPADE)

```python
class DummyAgent(Agent):
    class MyBehav(CyclicBehaviour):
        async def on_start(self):
            print("Starting behaviour . . .")

        async def run(self):
            print("Running the behaviour . . .")

    async def setup(self):
        print("Agent starting . . .")
        b = self.MyBehav()
        self.add_behaviour(b)

async def main():
    dummy = DummyAgent("your_jid@your_xmpp_server", "your_password")
    await dummy.start()

    await wait_until_finished(dummy)
```

Listing 2: A simple SPADE agent with a simple cyclic behaviour

Smart Python Agent Development Environment (SPADE)

Figure 3: Features of agent communication in SPADE

Smart Python Agent Development Environment (SPADE)

```python
class ReceiverAgent(Agent):
    class ReceiveBehav(OneShotBehavior):
        async def run(self):
            msg = await self.receive(timeout=10)
            if msg:
                print(f"Message received: {msg.body}")
            else:
                print("No message received.")

async def setup(self):
    print("Receiver Agent is starting...")
    self.add_behaviour(self.ReceiveBehav())
```

Listing 3: Implementing an agent that can receive messages.

Smart Python Agent Development Environment (SPADE)

```python
class SenderAgent(Agent):
    class SendBehav(OneShotBehavior):
        async def run(self):
            msg = Message(to="receiver@your_xmpp_server")
            msg.set_metadata("performative", "inform")
            msg.body = "Hello, Agent B!"
            await self.send(msg)
            print("Message sent!")

    async def setup(self):
        print("Sender Agent is starting...")
        self.add_behaviour(self.SendBehav())
```

Listing 4: Implementing an agent that can send messages.

MAGO

Developing a Framework for Agent Gamification Based on Ontologies (MAGO)

The result of a cooperation between:

- University of Zagreb Faculty of Organization and Informatics (UNIZG FOI) and
- Universitat Politècnica de València (UPV), Valencian Research Institute for Artificial Intelligence (VRAIN).

This cooperation is funded by the European Union and the Croatian Science Foundation.

Developing a Framework for Agent Gamification Based on Ontologies (MAGO)

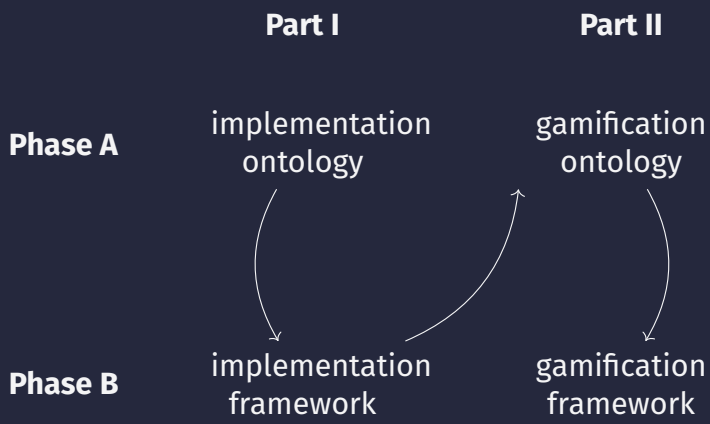

Figure 4: The flow between the parts and the phases

MAGO

MAGO-Ag Ontology

MAGO-Ag Ontology

- An ontology comprising concepts applicable to implementing multiagent systems (MASs) as intelligent virtual environments (IVEs).
- The main goal of the ontology is to enable the modelling of a multiagent system in terms of implementation possibilities.
- The ontology contains a selection of modified and enriched concepts of the MAMbO5 ontology, a result of earlier cooperation [2].
  - e.g. Agent, Behaviour, Action, Process, Objective, Artefact

MAGO-Ag Ontology

Figure 5: Visual relationship of the concepts of the MAGO-Ag ontology

MAGO

MAGO-Ag Framework

MAGO-Ag Framework

- The main objective of the MAGO-Ag framework is to translate a MAS modelled using the MAGO-Ag ontology into an implementation template for a MAS comprising SPADE agents.

MAGO-Ag Framework

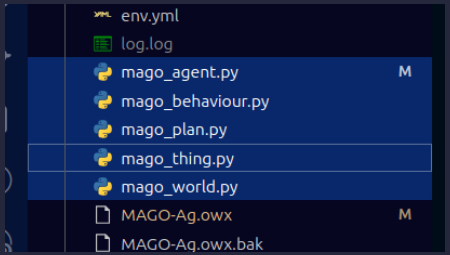

Figure 6: Essential files of the translation process

MAGO-Ag Framework

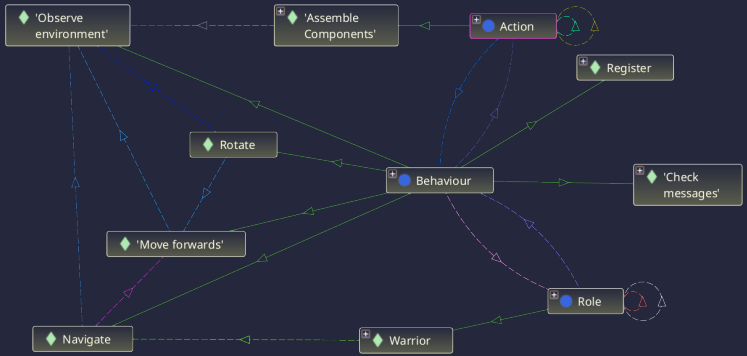

Figure 7: A selection of individuals modelling agent behaviour instance

MAGO-Ag Framework

```python
class Navigate(FSMBehaviour):

    async def on_start(self) -> None:
        print("Starting behaviour.")

    async def on_end(self) -> None:
        print("Ending behaviour.")

    async def state_setup(self):
        self.add_state(name='Observe_environment', state=Observe_environment(), initial=True)
        self.add_state(name='Move_forwards', state=Move_forwards())
        self.add_state(name='Rotate', state=Rotate())
        self.add_transition(source='Observe_environment', dest='Rotate')
        self.add_transition(source='Rotate', dest='Move_forwards')
        self.add_transition(source='Move_forwards', dest='Observe_environment')
```

Listing 5: Finite state machine behaviour implementation template with three state behaviours

MAGO-Ag Framework

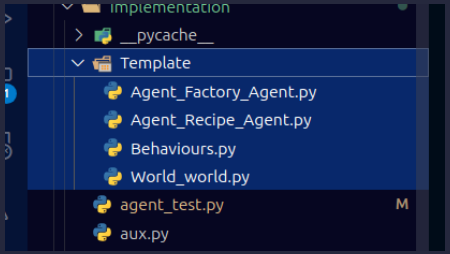

Figure 8: Generated template files for the modelled system

MAGO-Ag Framework

```
:~/$ python translate.py
. . .
:~/$ python Template/World_world.py
AgentAlice: New agent running.
AgentBravo: New agent running.
AgentClive: New agent running.
```

Listing 6: Running the translation script and the modelled system's generated implementation template

MAGO-Ag Framework

Figure 9: Pieces of knowledge available to agents after template generation

Conclusion

Conclusion

The framework, in its presented state, is a work-in-progress package.
Further improvements are seen in:

- rendering strategy-related concepts using languages that allow for reasoning;
- implementing the framework as a distributed system that would focus on deploying agent implementation templates over several workspaces;
- further testing the framework using different scenarios;
- adapting the implementation templates to different agent development frameworks.

Bibliography

Bibliography i

[1] S. J. Russell and P. Norvig, Eds., *Artificial Intelligence: A Modern Approach* (Pearson Series in Artificial Intelligence), 4th ed., in collab. with M.-w. Chang, J. Devlin, A. Dragan *et al.* Harlow, UK: Pearson Education Limited, 2022, 1166 pp., ISBN: 978-1-292-40113-3.

[2] B. Okreša Đurić, J. Rincon, C. Carrascosa, M. Schatten and V. Julian, 'MAMbO5: A new Ontology Approach for Modelling and Managing Intelligent Virtual Environments Based on Multi-Agent Systems,' *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 9, pp. 3629–3641, 2019, ISSN: 1868-5145. DOI: 10.1007/s12652-018-1089-4.

Acknowledgement

Acknowledgement

MOBODL-2023-08-5618
This project was funded by the European Union and the Croatian Science Foundation.

Bogdan Okreša Đurić
dokresa@foi.unizg.hr

University of Zagreb Faculty of Organization and Informatics Artificial Intelligence Laboratory

ai.foi.hr