ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS OF CESENA

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
Second Cycle Degree in Computer Science and Engineering

VISUAL PROGRAMMING PARADIGM FOR HYPERMEDIA MULTI-AGENT SYSTEMS

Thesis in
PERVASIVE COMPUTING

Supervisor
Prof. ALESSANDRO RICCI

Co-supervisors
Prof. SIMON MAYER
Dott. SAMUELE BURATTINI

Presented by
ALESSANDRO MARCANTONI

Academic Year 2021 – 2022

KEYWORDS

Multi-Agent Systems
Visual Programming
Organization
Hypermedia
Web IDE

To the ones who have always believed in me

Index

Introduction

Qui il testo dell'introduzione alla tesi. Generalmente l'introduzione non dovrebbe superare le 2/3 pagine e dovrebbe essere scritta solo alla fine.

Chapter 1

Context, Motivations and Research Proposal

This project was born thanks to the collaboration between the *Pervasive Software Lab*[1] of the *University of Bologna* and the *Interaction- and Communication-based Systems*[2] research group of the *University of St. Gallen*, in Switzerland.

Since both groups are interested and involved in similar research topics, such as the interactions among devices and people in ubiquitous computing environments, a highly enriching opportunity for an internship abroad arose.

Moreover, the research group in St. Gallen is contributing to a European project named *IntellIoT*[3], which stands for "Intelligent IoT" and whose aim is, together with its partners, to develop a reference architecture to enable IoT environments for semi-autonomous IoT applications endowed with intelligence that evolves with the Human-in-the-Loop. Hence, the concurring perspective of the two research groups and the St. Gallen group's participation in such a visionary initiative made it easy to shape the requirements of the thesis work.

The following sections will present the main objectives of the *IntellIoT* project to describe the context in which the thesis was conducted.

[1] https://apice.unibo.it/xwiki/bin/view/PSLab/
[2] https://ics.unisg.ch/chair-interactions-mayer/
[3] https://intelliot.eu/

1.1 The *IntellIoT* Project

IntellIoT is a Pan-European project that focuses on developing integrated, distributed, human-centered and trustworthy IoT frameworks, with particular attention to sectors like agriculture, healthcare, manufacturing, energy, construction, and smart cities.

To achieve the latter goals, *IntellIoT* explores and exploits new enabling technologies such as 5G connectivity, distributed technology, Augmented Reality, Artificial Intelligence, and tactile internet. Of course, this is possible thanks to the project's partners spread across ten countries that form a competitive ecosystem.

Among them, the *University of St. Gallen* is currently focusing on integrating physical things into the Web, increasing the autonomy of Web-enabled devices, and making interactions of connected devices intelligible for people using **Hypermedia Multi-Agent Systems**. Indeed, the primary objective of this thesis is to explore how humans can effectively define such systems' organization.

1.1.1 Mission

Smart technologies play a significant role in our life and work. However, the traditional approach based on cloud technologies has limitations, such as unreliable connectivity, limited bandwidth, long reaction times, lack of autonomy, and trust concerns. Therefore, the goal of *IntellIoT* is to tackle these issues and create a framework enabling next-generation IoT solutions. Specifically, these issues are addressed by focusing on the following three pillars, which are the central research topics of the project.

Collaborative IoT

Various semi-autonomous entities should cooperate to achieve the system's overall goal. Hence, self-awareness and knowledge of the task to perform and the environment where they are located are vital abilities to seek. Entities can acquire knowledge either by interacting with the environment via sensors or by reliable and secure communication with each other. However, since providing complete knowledge to entities in open and continuously changing environments is practically infeasible, Artificial Intelligence and Machine Learning come to the rescue.

Human-in-the-Loop

Since IoT applications cannot be completely autonomous in how they decide and act, humans need to be involved in controlling and optimizing the Artificial Intelligence the devices are endowed with. Indeed, the interaction between humans and intelligent systems can expand the latter's knowledge about the environment or the application by exploiting the former's experience. In fact, by applying Machine Learning algorithms, the devices can learn new features and information about the overall process so that they will have enough knowledge to react to similar scenarios in the future automatically.

Trustworthiness

As beneficial as IoT devices are, they present some major security concerns. The Mirai botnet exploiting embedded devices to perform DDoS attacks [2], possibly hackable cardiac devices, and Stuxnet sabotaging Iranian nuclear facilities [3] are only a few examples of critical breaches. Thus, security, privacy, and trust are vital for IoT systems and applications and their broader acceptance. Therefore, these concepts must be considered early in the design process and regard computation and communication infrastructure.

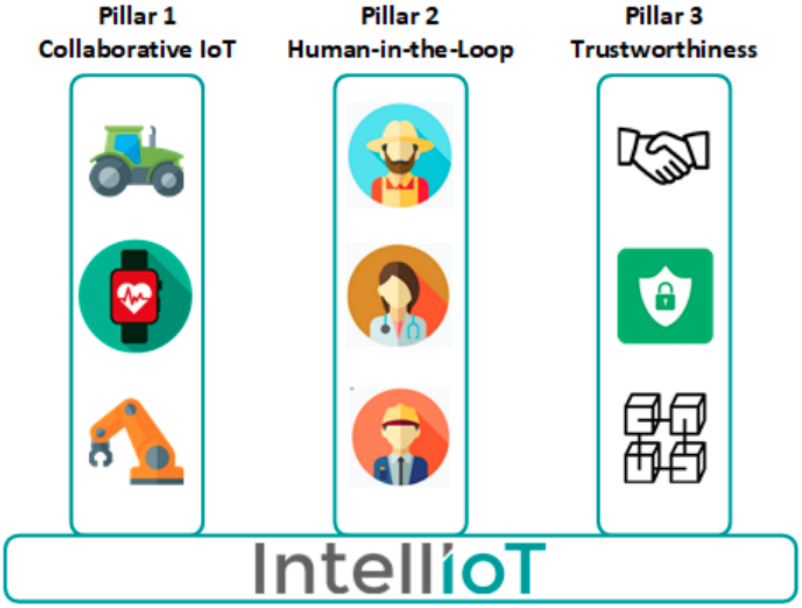

Figure 1.1: The three pillars of *IntellIoT*.

1.1.2 Use Cases

The above three key component areas are supported by *IntellIoT*'s dynamically managed network and computation infrastructure that, combined, provide resource and edge management, orchestration capabilities, and network choreography, exploiting cutting-edge technologies like 5G. Moreover, for the pillars to not remain only abstract concepts, various use cases that aim to address real-life problems in three core sectors were developed:

- **Agriculture**: the application of IoT in agriculture could be a life-changer for humanity as we now witness how extreme weather, deteriorating soil, dry lands, and collapsing ecosystems make food production more and more complicated and expensive, not to mention the population growth that increases the demand for resources. Although "smart farming" is already a thing, *IntellIoT* aims to bring it to the next level thanks to autonomous devices such as self-driving tractors and drones endowed with sensors and actuators. However, even though machines perform potentially dangerous, tiring, and repetitive tasks for humans, the latter still play a crucial role in managing the farm. Indeed, they can remote control the devices in uncertain situations, refining the Artificial Intelligence models. Additionally, human operators are in charge of defining the goals of the farming system, leveraging their experience and knowledge about the domain.

- **Healthcare**: IoT is revolutionizing the healthcare industry, mainly due to remote patient monitoring. Indeed, being endowed with specific sensors, devices can track the latter's vital signs and other health metrics and send the collected data to healthcare providers. Some examples of such devices are *Continuous Glucose Monitoring* [8] and *Dissolvable Brain Swelling*[13] sensors or ordinary smartwatches. This process improves the patient's quality of life, which does not need to be limited to their home or the hospital and can thus carry on with their regular everyday activities. Moreover, continuous monitoring and real-time data sharing allow timely interventions when necessary, and automatic data collection can drastically decrease the time and effort required to retrieve and manage information about the patient. Not to mention the opportunities for data analysis and possible Artificial Intelligence models that would support clinicians in being more efficient. However, this workflow potentially exposes patients' sensitive information; therefore, we find privacy, security, and trust assurance among the main focuses of *IntellIoT* regarding this use case.

- **Manufacturing**: IoT is one of the core driving forces behind Industry 4.0, which aims to digitalize and automate operational processes counting on as little support as possible from human operators, from the order submission to the delivery of the product. Leveraging AI and Machine Learning, robotics, and data analysis, IntellIoT envisions a future with shared manufacturing plants and multiple customers utilizing manufacturing as-a-service. According to the latter scenario, an intelligent IoT environment would derive a production plan from data received from a customer and select the appropriate machines for the planned steps. However, whenever AI is not sufficiently confident about a step, a human-in-the-loop can take over control remotely, providing information that will be learned by the Machine Learning algorithm thanks to continual learning.

1.2 Domain-Expert Programming

As already discussed, the crucial role of end-users in the definition of autonomous systems, the importance of their intervention in case of need, and their expertise are utterly unmatched by Artificial Intelligence algorithms and probably will be for a long time.

On top of that, the gap between programmers and end-users regarding domain comprehension is a well-acknowledged issue concerning software development. Indeed, developers' poor understanding of the domain often results in projects missing their schedules or exceeding their budget, poor-quality software, or even wrong functionalities [7]. To address this critical issue, several techniques have been developed. For instance, one of the core aspects of Domain Driven Design is *knowledge crunching*, which aims to extract domain knowledge from the experts to reflect it in the code during the subsequent development phases.

On the other hand, a different approach might be taken directly involving domain experts in the programming process. This kind of user can be defined as professionals in some domain different from computer science who need to use computers in their daily work and often have real needs to perform some programming activities that result in the creation or modification of software artifacts [6]. Given the latter definition and the premise suggesting the importance of domain expertise, providing domain experts with tools, such as domain-specific languages (DSL) or more user-friendly visual tools, that allow them to "code" or configure parts of complex systems feels very natural. Therefore the need for an intuitive development environment in which users with no proper computer science background can naturally and seam-

lessly transform their domain knowledge into specifications with low-code (or possibly no-code).

1.3 Proposing a Visual Programming Paradigm for Organizations

Multi-Agent Systems is one of the core enabling technologies of the infrastructure of *IntellIoT*. MAS fit IoT because they tackle the complexity and handle the decentralization of the IoT ecosystem by providing a framework for coordinating the actions of a large number of devices, allowing the latter to communicate and make decisions toward the achievement of a common goal. Another critical advantage of MAS is their ability to deal with partial knowledge, incomplete and imperfect information, and adapt and reason in real-time, which is crucial in dynamic, uncertain, open, and distributed networks.

However, the high-level expertise required to program agent-based systems hinders the large-scale adoption of MAS. Thus, efforts have been made to eliminate the entry barrier to MAS development, such as creating an agent-oriented programming paradigm [5], which enables individuals without coding experience, but with knowledge of specific target domains, to design and (re)configure autonomous software.

Nevertheless, although this block-based visual development environment is suitable for defining single entities, a level of abstraction to handle the relations among and coordination of the latter is still missing. Even though these aspects could be technically represented and managed directly in the mind of the agents, the adoption of adequate abstractions makes the solution dramatically more straightforward and elegant. [4]

Therefore, the idea is to design and implement a visual language and a supportive development environment for MAS organizations, which is a novelty to our knowledge. The core thesis work will be based on studying the existing organization models, with particular attention to the $\mathcal{M}$OISE$^+$ model, and developing an appropriate representation that could provide a suitable tradeoff between the expressiveness of code-based specifications and the ease of use of graphic programming interfaces. A qualitative evaluation is also planned to receive feedback for the developed framework and to study how non-technical users solve problems by exploiting the visual language.

The following chapter provides an introduction to the background of the main technologies and research topics explored in the project.

Chapter 2

Background

To better understand this thesis work, a brief excursus of the existing literature about the main concepts the project relies on is presented. The aim is to provide some basic knowledge of the topics to understand what is considered state-of-the-art and to introduce some of the technologies exploited to develop the project.

2.1 Multi-Agent Systems

2.1.1 What is an Agent?

The concept of a software agent can be traced back to the early days of research into Distributed AI in the 1970s when Carl Hewitt proposed the concurrent Actor model. In his paper, he introduced the concept of a self-contained, interactive, and concurrently-executing object which he termed "actor". The latter is a computational agent with a mail address and behavior. Actors can communicate by message-passing and carry out their actions concurrently. [9]

The term "agent" quickly spread to heterogeneous research fields; therefore, there is no commonly agreed definition for it. However, a generally accepted description of what an agent is is the following by Wooldridge [11]:

> *An agent is a self-contained program capable of controlling its own decision-making and acting, based on its perception of its environment, in pursuit of one or more objectives.*

To sum up, we can identify a set of features that an agent should possess:

- **Autonomy**: agents should be able to perform most of their tasks without the direct intervention of humans or other agents, and they should encapsulate control over their actions and internal state

- **Social ability**: agents should be able to interact with each other and possibly humans to complete their tasks.
- **Responsiveness** (situatedness): agents should perceive their environment and respond to changes in it.
- **Proactiveness**: agents should exhibit opportunistic, goal-directed behavior and take the initiative when appropriate.

During the first years, the research concentrated on interaction and communication among agents, decomposition and distribution of tasks, and coordination and cooperation. The goal was to specify, analyze, design, and integrate systems containing multiple collaborative agents.

2.1.2 From the Individual to the Collective

Multi-Agent Systems have been studied as a per se field since the 1980s and gained widespread recognition in the 1990s. Since then, international interest in the topic has grown enormously as agents are considered an appropriate paradigm to exploit the possibilities presented by massive open distributed systems. Moreover, MAS seem to be a natural metaphor for understanding and building a wide range of what we might call *artificial social systems*. [17]

According to the *Alan Turing Institute* [10]

> *A Multi-Agent System consists of multiple decision-making agents which interact in a shared environment to achieve common or conflicting goals.*

As it can be noticed in fig. 2.1, MAS are composed of an environment and agents existing within it that are bonded by relations.

Environment

The environment in MAS plays a dual role [16]:

- *The "external world".* Agents become aware of the context they are immersed in and its dynamics by perceiving the environment through sensors. Moreover, they pursue their goals through actions performed by actuators that aim at modifying the environment, eventually reaching the latter's desired state.
- *A medium for coordination.* Agents exploit the environment to share information and coordinate their behavior. Each agent follows simple behavioral rules, resulting in a collective behavior that is more complex

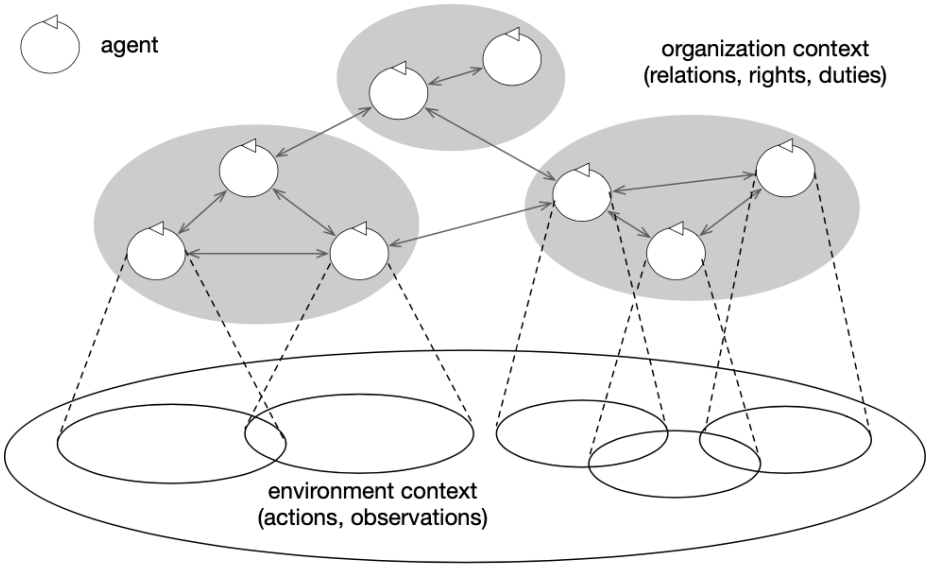

Figure 2.1: A representation of Multi-Agent Systems. [12]

than the sum of the individual behavior; this pattern resembles stigmergic systems in which agents coordinate their behavior through the manipulation of marks.

The environment not only enables the agents to interact with the deployment context, giving them access to sensors and actuators but also provides them with external resources that they can exploit to achieve their goal.

The *Agents & Artifacts* (A&A) conceptual framework [14] argues that, just like in human society, MAS environments should contain different kinds of objects, tools, and artifacts in general that agents can use to support their activities. This vision also constitutes a revolution from an engineering perspective as it encourages system designers to model the environment as a set of *artifacts*, each of which encapsulates its intended purpose and exposes its observable state. Moreover, the A&A meta-model provides an effective abstraction level that shields low-level details of the deployment context, so that designers can focus on the agents' behavior.

Organization

There are two approaches to MAS engineering. [1] The first one regards *agent-centered* MAS, in which the focus is given to individual agents. According to this viewpoint, the designer should concern about the local behavior of the agents and their interactions without worrying about the global structure

and goal of the system as the latter should emerge as a result of the lower-level individual interactions in a bottom-up fashion. The key issues of this approach are unpredictability and uncertainty since it could lead to undesirable emergent behaviors. As Weyns [15] stated, giving the responsibility of system organization implicitly to individual agents is highly complex and not suitable for real-world large-scale scenarios.

The second approach is *organization-centered* MAS, in which the structure of the system is given greater attention. The developer designs the entire organization and coordination patterns on the one hand, and the agents' local behavior on the other. This can be seen as a top-down approach as the organization abstractions impose constraints on the agents and regulate their interactions, simplifying the design of complex and scalable systems and allowing more accurate modeling of the problems being tackled.

2.2 Organizations in Multi-Agent Systems

2.3 Hypermedia Multi-Agent Systems

Chapter 3

Requirements

Chapter 4

Design

4.1 From Organization to a Visual Language

4.1.1 Organization Structure

4.1.2 Collective Goals

4.1.3 Norms

4.1.4 Organization Instance

4.2 Architecture and Main Components

4.2.1 *ORA4MAS* Organizational Artifacts

Chapter 5

Development

5.0.1 Web IDE

5.0.2 Backend and Specifications Storage

5.0.3 Running Organization Entities

Chapter 6

Evaluation

6.1 Case Study - Smart Farming

6.2 Outcome Analysis

Conclusions

Qui il testo delle conclusioni alla tesi. Non deve essere un riepilogo di quanto fatto nella tesi ma piuttosto le conclusioni raggiunte relative al lavoro svolto.

Acknowledgments

Qualora lo si desideri è possibile inserire qui il testo dei ringraziamenti alle persone che hanno contribuito in qualche modo al percorso che ha portato al lavoro di tesi.

Bibliography

[1] Hosny Ahmed Abbas. Organization of multi-agent systems: An overview. *International Journal of Intelligent Information Systems*, 4(3):46, 2015.

[2] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. Understanding the mirai botnet. In *26th USENIX security symposium (USENIX Security 17)*, pages 1093–1110, 2017.

[3] Marie Baezner and Patrice Robin. Stuxnet. Technical report, ETH Zurich, 2017.

[4] Olivier Boissier, Rafael H Bordini, Jomi Hubner, and Alessandro Ricci. *Multi-agent oriented programming: programming multi-agent systems using JaCaMo*. MIT Press, 2020.

[5] Samuele Burattini, Alessandro Ricci, Simon Mayer, Danai Vachtsevanou, Jérémy Lemee, Andrei Ciortea, and Angelo Croatti. Agent-oriented visual programming for the web of things. 2022.

[6] Maria-Francesca Costabile, Daniela Fogli, Catherine Letondal, Piero Mussio, and Antonio Piccinno. Domain-expert users and their needs of software development. In *HCI 2003 End User Development Session*, 2003.

[7] Salem Ben Dhaou Dakhli and Mouna Ben Chouikha. The knowledge-gap reduction in software engineering. In *2009 Third International Conference on Research Challenges in Information Science*, pages 287–294, 2009.

[8] Andrea Facchinetti, Giovanni Sparacino, Stefania Guerra, Yoeri M Luijf, J Hans DeVries, Julia K Mader, Martin Ellmerer, Carsten Benesch, Lutz Heinemann, Daniela Bruttomesso, et al. Real-time improvement of continuous glucose monitoring accuracy: the smart sensor concept. *Diabetes care*, 36(4):793–800, 2013.

[9] Carl Hewitt. Viewing control structures as patterns of passing messages. *Artificial intelligence*, 8(3):323–364, 1977.

[10] Alan Turing Institute. Multi-agent systems. Accessed January 11, 2023 `https://www.turing.ac.uk/research/interest-groups/multi-agent-systems`.

[11] N. Jennings and M. Wooldridge. Software agents. *IEE Review*, 42(1):17–20, 1996.

[12] Nicholas R Jennings. On agent-based software engineering. *Artificial intelligence*, 117(2):277–296, 2000.

[13] Seung-Kyun Kang, Rory KJ Murphy, Suk-Won Hwang, Seung Min Lee, Daniel V Harburg, Neil A Krueger, Jiho Shin, Paul Gamble, Huanyu Cheng, Sooyoun Yu, et al. Bioresorbable silicon electronic sensors for the brain. *Nature*, 530(7588):71–76, 2016.

[14] Alessandro Ricci, Mirko Viroli, and Andrea Omicini. Give agents their artifacts: The a&a approach for engineering working environments in mas. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '07, New York, NY, USA, 2007. Association for Computing Machinery.

[15] Danny Weyns, Robrecht Haesevoets, and Alexander Helleboogh. The macodo organization model for context-driven dynamic agent organizations. *ACM Trans. Auton. Adapt. Syst.*, 5(4), nov 2010.

[16] Danny Weyns, Andrea Omicini, and James Odell. Environment as a first class abstraction in multiagent systems. *Autonomous agents and multi-agent systems*, 14(1):5–30, 2007.

[17] Michael Wooldridge. *An introduction to multiagent systems*. John wiley & sons, 2009.