information

MDPI

Article

The Impact of the Organization on the Autonomy of Agents

Zouheyr Tamrabet [1,2], Djamel Nessah [3], Toufik Marir [1], Varun Gupta [4,5,*] and Farid Mokhati [1]

[1] Mathematics and Computer Science Department, Research Laboratory on Computer Science's Complex Systems, University of Oum El Bouaghi, Oum El Bouaghi 04000, Algeria; zouheyr.tamrabet@ensia.edu.dz (Z.T.); marir.toufik@univ-oeb.dz (T.M.); mokhati.farid@univ-oeb.dz (F.M.)
[2] Intelligent Systems Engineering Department, The National School of Artificial Intelligence, Mahelma 16000, Algeria
[3] ICOSI Laboratory, Computer Science Department, University of Khenchela, Khenchela 40000, Algeria; nessah_djamel@univ-khenchela.dz
[4] Multidisciplinary Research Centre for Innovations in SMEs (MrciS), Gisma University of Applied Sciences, 14469 Potsdam, Germany
[5] Department of Economics and Business Administration, University of Alcala, 28801 Alcalá de Henares, Spain
* Correspondence: varun.gupta@uah.es

Abstract

In multi-agent systems (MAS), autonomy is a fundamental characteristic that enables agents to operate independently and adaptively within complex environments. However, such characteristics may cause the system to fall into undesirable situations. On the one hand, purely autonomous agents are difficult to predict. On the other hand, fully controlled agents lose many of their abilities. Therefore, control frameworks have been designed in the form of organizational architectures to help address the need for balance between purely autonomous and fully controlled agents. This paper investigates the impact of organization on the autonomy of the agents. To measure this impact, we propose a set of seven metrics (Behavioral Wealth (BW), Service Wealth (SW), Frequency of Service Searches per Time (FoSST), Frequency of Service Searches per Behavior (FoSSB), Number of Service Searches (NoSS), Number of Service Demands per Behavior (NoSDB), and Number of Provided Services per Demand (NoPSD)) and apply them to a case study implemented in two configurations: with and without organizational aspects. To model organizational aspects, we adopt the Agent–Group–Role (AGR) model, chosen for its structured approach to defining agent responsibilities and interactions. The findings of this study show that the organizational aspects reduce the communication load and enhance the effectiveness of agents.

Keywords: multi-agent systems; organization; autonomy; measures; organizational models; organizational impact

check for updates

Academic Editor: Christoforos Kachris

Received: 21 August 2025
Revised: 12 September 2025
Accepted: 25 September 2025
Published: 27 September 2025

Citation: Tamrabet, Z.; Nessah, D.; Marir, T.; Gupta, V.; Mokhati, F. The Impact of the Organization on the Autonomy of Agents. *Information* **2025**, *16*, 838. https://doi.org/10.3390/info16100838

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The multi-agent systems (MAS) paradigm has significantly influenced our perspective on software. This paradigm has arisen at the intersection of various fields, including distributed systems, artificial intelligence, and software engineering. As a result, software within this paradigm is conceptualized as a collection of interacting intelligent entities known as agents. An agent is a logical or physical entity that seeks to achieve its own goals [1]. The distinction between an agent and other traditional software entities is established through several characteristics, of which the most important are situation, reactivity, proactivity, sociability, and autonomy [2]. Among these characteristics, autonomy is fundamental.

Although autonomy is a key concept in many recent systems [3], an excess of autonomy risks bringing the entire system into undesirable situations. On the one hand, since the agents are all part of the same system, fully autonomous behaviors may endanger their common goals. On the other hand, entirely controlled behaviors deteriorate the agents' capabilities. Considering multi-robot systems working to explore an unknown environment (e.g., a Mars planet). It is difficult to directly control all the robots due to the unpredictable nature of the environment [4]. Moreover, it is important to ensure a level of control that enables better coordination among the robots in order to avoid chaotic situations, such as contradictory behaviors. Consequently, several solutions have been proposed in the literature to address this issue [5–7]. Among these solutions, controlling the agents through organizational mechanisms allows them to behave in a balanced way, avoiding both fully controlled and fully autonomous behaviors [8].

Organization is one of the main concepts in multi-agent systems. A multi-agent system can then be defined as an organized set of agents [9]. An organization describes how the members of the organization interact to achieve a common goal. However, the importance of organizations for multi-agent systems cannot be limited to the simple structuring of the system. Organization increases system efficiency by decreasing the search space that an agent needs to consider to accomplish its objective. In addition, it improves modularity and reusability in the design of MAS by using generic diagrams (for example, abstract roles) [8].

Moreover, to establish a solid organization, it is necessary to use an organizational model. An organizational model is a simplified representation that specifies the structure and functionalities of an agent society. Several models have been proposed for the specification of multi-agent organizations. The AGR (Agent–Group–Role) model [10] falls within the scope of these proposals. As shown in Figure 1, this model is based on a simple principle: "each agent plays at least one role and belongs to one or more groups." In addition to the simple nature of the AGR model that encourages its use even for non-specialists, this model has undergone several extensions to include the environment [11], the service [12], fuzzy logic and principles of the Markov Decision Process (MDP) [13], and norms [14]. Moreover, this model has been used as a foundation for the development of multi-agent systems in several works [15–18].

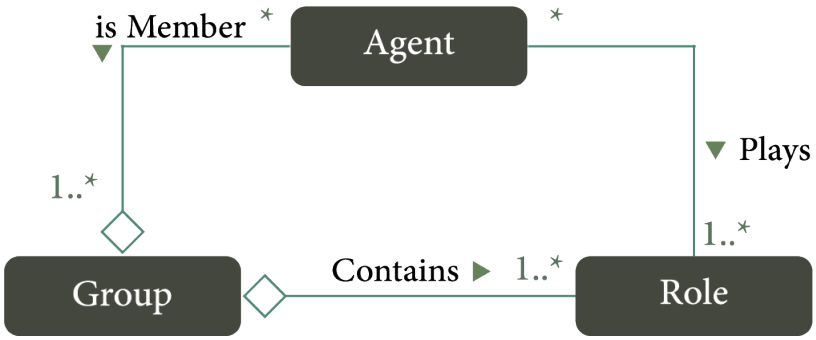


Figure 1. The AGR model.

Our goal is to estimate the impact of organization on the autonomy of agents. Hence, we proposed several metrics to evaluate the autonomy of agents. By proposing these metrics, we enable the assessment of different dimensions of agent autonomy. In fact, we have theoretically analyzed the impact of the organization on each metric, which allows us to better understand its influence on autonomy. Hence, we have empirically studied this impact by applying the proposed metrics to two different systems, one developed with organization and the other without. The AGR model was adopted as the organizational framework for developing the multi-agent system for the reasons discussed above.

By understanding the impact of organization on autonomy, we can benefit from several advantages. First, this allows the use of an organizational model to control agents indirectly, rather than attempting to control them directly, which may otherwise constrain their capabilities. Moreover, from an engineering perspective, this approach facilitates mastering the development of agent-based applications. In fact, the proposed metrics can be employed to estimate the efficiency of the developed application (e.g., through the number of exchanged messages or the search time), thereby providing insights into whether the organizational model improves this characteristic or not.

We then applied these metrics to two different systems developed with and without organization. The organizational model we used to develop the multi-agent system with organization is AGR, for the reasons cited above.

The remainder of this paper is organized as follows. Section 2 presents related works to our study. Section 3 introduces the proposed metrics. Section 4 presents the measurement tool, highlighting its modular design and implementation. Section 5 describes the case study used to validate our approach, while Section 6 discusses the results and their implications. Finally, Section 7 concludes the paper and outlines potential directions for future work.

2. Related Works

Multi-agent systems, as a relatively new software paradigm, have introduced several concepts such as autonomy, organization, and the environment. Usually, researchers study each of these concepts separately. In this paper, our goal is to study the impact of organization on autonomy. Hence, we devote this section to presenting the most relevant studies on both concepts.

2.1. Research on Agent Autonomy

Autonomy represents the main characteristic of agents. However, there is no commonly agreed definition of what it is. Hence, many works have focused on the specification and measurement of agent autonomy from an empirical point of view [9,19–21]. To clarify the ambiguity associated with early works targeting the agency and autonomy concepts, Luck & d'Iverno [19] proposed a formal specification of these concepts using the Z language. Thus, agency is an object with a goal, while an autonomous agent is an agent with motivation. Instead of ordinary goals, which are adopted from other agents, a motivation is a goal generated by the agent itself.

Based on the same principle, namely the ambiguity of agent autonomy, Barber & Martin [20] proposed to specify and measure the autonomy of agents. They progressively discussed definitions of autonomy to reach a comprehensive one. According to their point of view, autonomy can be measured as a degree rather than an absolute value. It is related to a given goal and the capabilities used to achieve it, and it represents the degree of freedom the agent possesses during the decision-making process. The authors then proposed measuring agent autonomy based on goals, the decision-making framework, and authority constraints.

In the same context, ref. [9] proposed several measures to quantify autonomy. The authors distinguished two kinds of autonomy: action autonomy and decision autonomy. Decision autonomy is measured by preference-autonomy and choice-autonomy metrics. They also measured the autonomy of an agent according to agent–user interaction and the group of agents.

To evaluate the autonomy of software agents, Alonso et al. [21] decomposed the autonomy concept into three attributes: self-control, functional independence, and evolution capabilities. Each of these attributes is measured by several metrics. For example, evolution

capabilities are measured by two metrics: the State Update Capacity and the Frequency of State Update.

From a practical standpoint, autonomy is not a static concept to be initially established. Instead, the level of an agent's autonomy should be adapted to real situations during runtime. Hence, several studies target adjustable autonomy [5,22,23]. In ref. [22], agents should dynamically adjust their interaction during decision-making to the runtime conditions. This kind of adjustment takes place at the organizational level and represents a type of organizational restructuring where changes in the control relationships between agents are permitted while the system is still in action. It is important to note that dynamic adaptive autonomy not only enables agents to adjust their level of autonomy but also enhances the adaptability and efficiency of multi-agent systems. On the other hand, Scerri et al. [23] proposed an adjustable autonomy approach based on Markov Decision Processes (MDPs) for decision-making control transfer.

In the same context of adjustable autonomy, Mostafa et al. [5] proposed a fuzzy logic-based adjustable autonomy framework that quantitatively measures and distributes autonomy among operators and dynamically adjusts their autonomy to match their ability to perform and produce desirable outcomes. The application of this framework introduces a deeper dimension to the autonomy of the agents by capturing relationships between autonomy attributes and managing adjustable autonomous multi-agent systems.

Despite autonomy being an individual attribute of an agent that enables decision-making, it is intricately linked to the collective aspects of a multi-agent system. Indeed, the social level facilitates distributed decision-making. This relationship is underscored in studies that adjust the level of autonomy based on organizational situations [22]. Goodrich et al. [24] also demonstrate that larger groups of agents have a higher success rate compared to individual agents while exploring trajectories toward a common goal, and emphasize the interaction algorithms that help many agents communicate and coordinate with one another. They study autonomy by introducing two novel dimensions: behavior potential and success potential. Consequently, they present an interesting viewpoint on autonomy and its role in the development of multi-agent systems.

Although the fundamental notions of the agent paradigm have been studied for more than three decades, there are still works that explore new dimensions of this paradigm. For example, Dodig-Crnkovic and Burgin [25] proposed a systematic classification of autonomous agents, inspired by natural systems, to clarify existing agent types and anticipate future specialized ones. Their framework complements empirical studies and situates autonomy in a broader context that also relates to organizational aspects.

Finally, autonomy has emerged as the essential characteristic that motivates researchers to integrate agents into Large Language Models (LLMs) in the current revolution of generative artificial intelligence [26–28].

2.2. Research on Multi-Agent Organizations

In addition, our study focuses on multi-agent organizations as a main concept in this field. One of the key challenges in multi-agent systems is how to effectively organize and manage the behavior of autonomous agents to achieve specific goals. To address this challenge, researchers have developed a range of organizational structures and mechanisms for controlling agent behavior, such as the Contract Net Protocol, auction-based mechanisms, and Holonic Multi-Agent Systems [29,30]. These approaches enable agents to interact and negotiate with one another in ways that support the efficient allocation of tasks and resources.

Given the importance of the self-organization concept in controlling complex systems, Boes & Migeon [31] proposed a framework for self-organizing multi-agent systems that

includes three levels: the individual-agent level, the team level, and the global level. At the individual-agent level, agents use local information and decision-making rules to perform their tasks. At the team level, agents coordinate their actions to achieve shared goals. Finally, at the global level, the system adapts to changes in the environment and the goals of the system. The authors claimed that self-organization can provide a more effective and efficient approach to controlling complex systems compared to traditional centralized approaches. Even though this work presents a good model for self-organizing multi-agent systems, there is limited empirical validation of the proposed approach.

By exploring the specialized literature, several studies targeting the self-organizing concept can be identified. For instance, Lhaksmana et al. [32] proposed a role-based approach to designing agent behavior in self-organizing multi-agent systems. The authors contend that traditional approaches to designing agent behavior, such as rule-based or goal-based methods, may not be suitable for self-organizing systems, which require agents to adapt their behavior based on the changing environment and their roles within the system. They propose a role-based approach that focuses on defining the roles agents can assume within the system and the associated behaviors and interactions. Although the article provides a useful framework for designing agent behavior, it does not address important aspects of self-organizing multi-agent systems, such as coordination, communication, and adaptation.

Recently, self-organization has been studied through the application of machine learning techniques. As an illustration, Wang et al. [33] suggested a novel method for multi-agent reinforcement learning (MARL) that incorporates the concept of emergent roles. They developed the Roma framework, which enables agents to evolve and acquire new roles based on interactions with other agents and their environment. The Roma framework is designed to address the challenges of MARL in large-scale and complex environments. It includes a role discovery mechanism that enables agents to identify their roles and a role management mechanism that allows them to switch roles or collaborate with other agents to achieve their objectives. Role discovery is based on a modified version of the Q-learning algorithm that assigns roles to agents according to their behavior and interactions. The results reported in this paper are very promising. However, challenges related to scalability remain unresolved.

2.3. The Relationship Between Autonomy and Organization

Autonomy and organization are closely related concepts [34,35]. Several organizational aspects (such as organizational structure, coordination techniques, and organizational rules or social norms) allow the control of agents [29,30,34].

In fact, an organization provides the schemes of relationships between agents and the coordination mechanisms that regulate their interactions. There are several types of relationships between agents, such as authority, dependence, collaboration, and competition. Consequently, some relationships limit the autonomy of agents by forcing them to act according to the requirements of others. Similarly, coordination mechanisms (such as interaction protocols) define a general interaction scheme that agents must follow in order to achieve a common goal. Adhering to this scheme naturally constrains the autonomy of agents [29].

A survey by Cortés and Egerstedt [4] highlights how the organization of multi-robot systems plays a central role in enabling coordinated and autonomous behaviors. By focusing on decentralized and scalable control strategies, the work shows how local interaction rules between robots can give rise to global, emergent properties such as formation control, coverage optimization, boundary tracking, and flocking. These coordination mechanisms rely on limited information exchange and local sensing, yet they allow teams of robots to autonomously reach collective goals without centralized supervision. This perspective il-

lustrates how organizational structures and interaction protocols, when properly designed, can balance autonomy at the agent level with the constraints necessary to achieve coherent group-level behaviors.

In the same context, multi-agent organizations introduce norms as a fundamental concept [36]. Norms represent rules that agents are expected to follow. Research in normative multi-agent systems has proposed several mechanisms to ensure agents' compliance with norms, such as regimenting and enforcement. Obviously, some mechanisms give agents more autonomy (such as enforcement) by allowing them to execute prohibited actions under the risk of punishment. In contrast, regimenting mechanisms prevent agents from executing prohibited actions, thereby limiting their autonomy.

Normative aspects also allow reasoning about norms in an autonomous way. Pacheco [34] proposed a model that enables agents to ensure compliance with norms without the use of regimenting mechanisms. In this model, the expected behavior and the corresponding punishments are specified using deontic and action logics. Based on these specifications, agents can reason and behave autonomously in accordance with norms, without direct control.

Moreover, in their study, Schillo & Fischer [35] clarified the relationship between autonomy classes and the design parameters of an organization. In fact, the authors distinguished several classes of autonomy, such as norm autonomy, exit autonomy, and representational autonomy. Each of these classes is then related to a design decision that must be made during the design of a holonic organization.

According to the specialized literature [8,29,30], it is widely acknowledged that organization has a significant influence on agents' autonomy, a fact that is often considered intuitive. However, despite this recognition, no empirical studies have systematically examined this relationship. To address this gap, we propose a set of metrics designed to evaluate the impact of organizations on autonomy. These metrics are applied to a case study implemented both with and without organizational mechanisms in order to provide a comparative assessment.

3. Proposed Metrics

Proposing metrics to measure autonomy should be based on a clear definition of the autonomy concept. As previously explained, there is no common definition of this concept. The diversity of the autonomy concept is due to the variety of perspectives (Is autonomy related to action or decision? Is autonomy viewed by the designer, the users, or the other agents?) [9] and the variety of its types [35,37,38].

To ensure consistency in our approach, we chose to base this work on a minimalist definition of autonomy. Hence, autonomy represents the ability of an agent to work without external intervention or control. We believe this definition encompasses all the attributes of the concept. Other characteristics, such as internal motivation or the evolution of capabilities, are dependent on additional features (such as the proactivity and learnability of the agent, respectively). Also, some definitions of autonomy are related to implicit characteristics, such as being goal-oriented, which makes measuring them objectively very difficult.

Based on this definition, autonomy is related to three components: the agent, the behavior, and the service. Behavior represents the capabilities of the agent, while service represents the dependency relationship between agents. An agent may require the intervention of other agents by requesting the accomplishment of a service. Consequently, the proposed metrics to measure autonomy are as follows:

3.1. Behavioral Wealth (BW)

Logically, an agent with more behaviors is considered to have more skills to achieve its objectives. Accordingly, an agent with the maximum number of behaviors has a higher autonomy rate than the others. This metric therefore represents the *Behavioral Wealth of an agent (A)*. As presented in Equation (1), we calculate the number of behavior classes implemented within an agent (Number of Behavior classes of agent A) compared to the total number of behaviors declared (Number of Total Behaviors).

$$BW_A = \frac{Number\ of\ Behaviors\ Classes_A}{Number\ of\ Total\ Behaviors} \tag{1}$$

3.2. Service Wealth (SW)

Considering that an agent has a set of services, the number of its services represents its skills. An agent with a higher number of services can achieve its objectives with minimal interaction with other agents. These interactions are, in fact, due to the agent's lack of certain services. This absence forces the agent to interact with others, which diminishes its autonomy. *Service Wealth (SW)* is therefore defined as the number of services provided by an agent compared to the total number of services provided by all agents (Equation (2)). This metric is dynamic because it can change over time.

$$SW_A = \frac{Number\ of\ Service_A}{Number\ of\ Total\ Services} \tag{2}$$

3.3. Frequency of Service Searches per Time (FoSST)

Sometimes, during execution, an agent searches for other agents offering particular services. This search indicates that it needs the help of other agents to continue its activity. The frequency of service searches within a time unit is a ratio that reflects the lack of an agent's services during that period (Equation (3)).

$$FoSST_A = \frac{Number\ of\ Service\ Search_A}{\Delta T} \tag{3}$$

3.4. Frequency of Service Searches per Behavior (FoSSB)

Sometimes, the frequency of service searches within a time unit does not provide useful information for MAS developers. This metric, when considered over the time unit, gives a global view of the agent's autonomy. On the other hand, MAS developers (such as the maintenance team) seek to identify the behaviors responsible for the increase in service searches. Identifying these behaviors allows determining the causes that decrease autonomy. Therefore, we propose the frequency of service searches per behavior executed (Equation (4)). The smaller this metric, the higher the autonomy. This metric is also dynamic.

$$FoSSB_A = \frac{Number\ of\ Service\ Search_A}{Number\ of\ Executed\ Behaviors_A} \tag{4}$$

3.5. Number of Service Searches (NoSS)

This metric specifically targets services and represents the number of searches for a service by all agents. Using this metric, we can identify the most frequently searched service. Consequently, we can enhance the autonomy of agents by embedding this service within the agents.

$$NoSS_S = [Number\ of\ (S)\ Search/Agents] \tag{5}$$

3.6. Number of Service Demands per Behavior (NoSDB)

Searching for a service is only one step in requesting services. Usually, agents follow the search step by requesting the service. Requesting a service can be critical in the sense that the executed behavior may not reach its goal. On the other hand, thanks to the autonomy of the agent, it can continue its execution and reach its goal without receiving the requested service. Hence, this metric (Equation (6)) represents the number of service demands compared to the number of achieved behaviors. Service demands are calculated based on the number of REQUEST and CFP messages.

$$NoSDB_A = \frac{Number\ of\ Service\ Demands_A}{Number\ of\ Achieved\ Behaviors_A} \tag{6}$$

3.7. Number of Provided Services per Demand (NoPSD)

When an agent receives a service demand, it can respond positively by providing the required service or negatively by refusing. Providing the requested service helps other agents reach their goals. On the other hand, refusing to provide the requested service implies that other providers in the multi-agent system must compensate to address the missing services. Equation (7) defines this metric as the ratio of provided services to the number of service demands.

$$NoPSD_A = \frac{Number\ of\ Provided\ Services_A}{Number\ of\ Service\ Demands_A} \tag{7}$$

Knowing that our metrics are essentially based on the concept of service, and following a thorough analysis of the different metrics, we can make the following assumptions:

- The two metrics, *Behavioral Wealth (BW)* and *Service Wealth (SW)*, will not be affected by the development paradigm of the system (with or without organization). In fact, the number of behaviors in each agent will be the same in both cases, with only minor differences due to additional behaviors for managing the organization (in the case of developing a multi-agent system with organization). Similarly, the services provided by the system will not be affected by whether it is implemented with or without organization.
- The metrics related to searching services—*Frequency of Service Searches per Behavior (FoSSB)*, *Frequency of Service Searches per Time (FoSST)*, and *Number of Service Searches (NoSS)*—will be reduced in systems with organization, because the organization allows agents to know the services provided by others without searching.
- The metrics related to the effectiveness of service requests—*Number of Service Demands per Behavior (NoSDB)* and *Number of Provided Services per Demand (NoPSD)*—will be improved when implementing a system with organization, because the authority relationships between agents will limit the refusal of service.

4. Measurement Tool

In our work, we propose a set of dynamic and static metrics. Static metrics are calculated directly from the source code of the system under measurement. In contrast, dynamic metrics require the execution of the system and are computed using aspect-oriented programming, as suggested by Marir et al. [39].

Aspect-oriented programming (AOP) [40] is a software paradigm proposed to improve software modularity, thereby facilitating development and maintenance. For this purpose, it separates core functionalities from transversal concerns (often non-functional). Transversal concerns are developed independently of the system and are automatically integrated into it by means of the weaver. The basic concepts of AOP include ***join points***, which

represent well-defined points in the program and can be either the execution or the call of a method; ***pointcuts***, which define sets of join points where additional behavior should be applied; and ***advice***, which specifies the additional code to be executed at those join points. The advice code can be executed *before*, *after*, or *around* a joinpoint. Figure 2 presents an example of an aspect developed using AspectJ [41]. In this example, an aspect called *MainAspect* is defined. Within this aspect, a pointcut (*MyMainPoint*) is declared to intercept the execution of the `main` method. The advice specifies that certain instructions will be executed *before* the execution of the `main` method.

```
public aspect MainAspect {
    pointcut MyMainPoint() : execution(* main(..));
    before() : MyMainPoint() {
        // launch the app
        // collect event info
        // Send event to server
    }
}
```

Figure 2. An example of an aspect code.

As presented in Figure 3, the tool developed to measure the proposed metrics consists mainly of a library of aspects that are employed to calculate the dynamic metrics. These aspects are developed using AspectJ [41], which is an extension of the Java programming language that supports aspect-oriented programming. Since the developed tool is intended to measure the metrics of JADE-based applications, and given that JADE [42] is itself an extension of Java to support the development of multi-agent systems, JADE [42] and AspectJ [41] can be naturally integrated. Hence, when the application under measurement begins its execution, the aspects intercept the execution of the main events that influence the metrics and store them in a dedicated data structure. The stored events, which represent the execution scenario, are then used to calculate the dynamic metrics according to the previously defined formulas. In contrast, static metrics are calculated directly from the source code of the application. Finally, the results of the metrics can be presented in graphical form to facilitate their interpretation.

It is important to note that we saved the intercepted events rather than calculating the metrics on the fly during execution, in order to avoid any impact of metric computation on the JADE-based applications. The intercepted events include behavior executions, communication acts (sending and receiving messages), and service searches, among others.

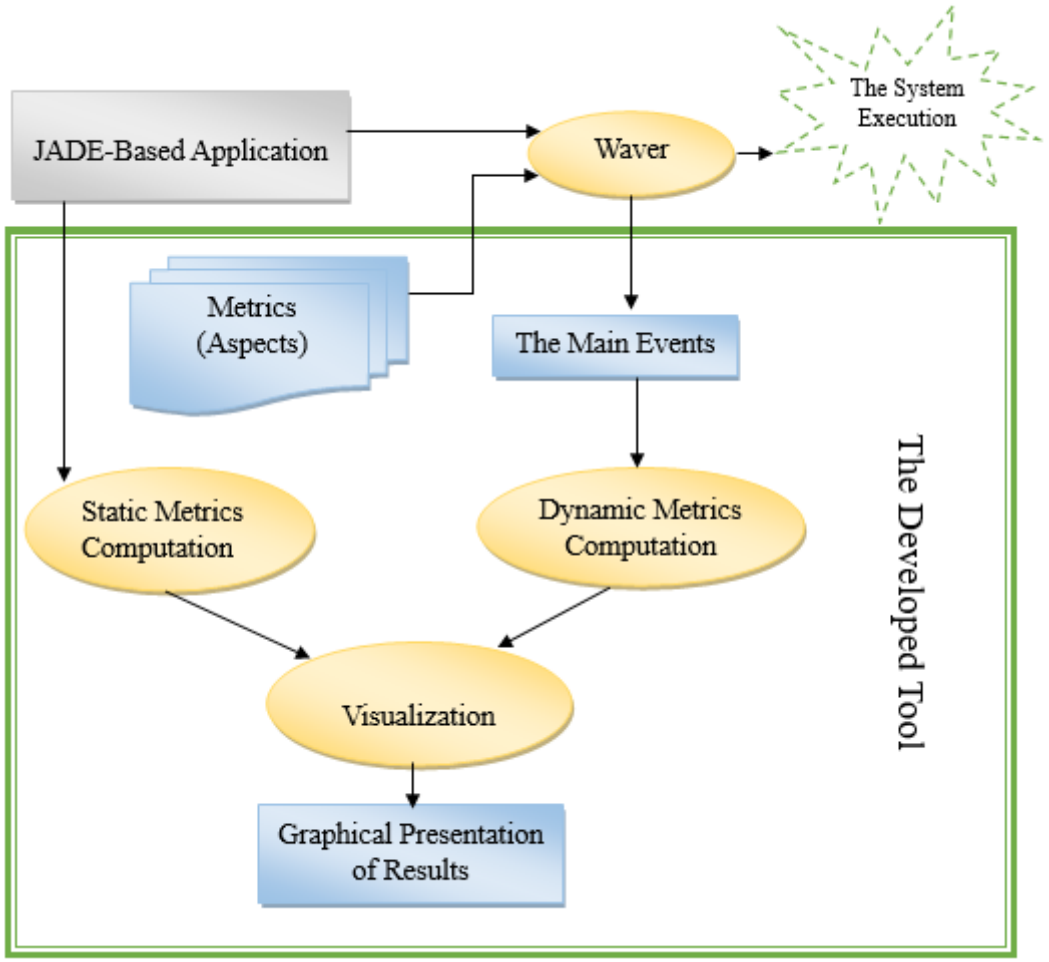


Figure 3. Overall architecture of the developed measurement tool.

5. Case Study

To demonstrate how organization in multi-agent systems affects the autonomy of agents, we applied the proposed metrics to a case study implemented in two different versions. The first version is without an explicit organization. The second version is developed according to the AGR organizational model [10]. The case study represents teacher assignments for a course and is implemented using the JADE platform [42].

The first example depicts a situation without an organization. In this case, the agents are teachers, each qualified with a set of services (the courses they can teach). A teacher makes perceptions once an event is captured (an event in this case represents a new assignment, characterized by a day, a time, and a type of service). The teacher verifies whether the event is compatible with its competencies. If it is qualified for the requested service, it also checks its timetable. If it is available, it accepts the request and becomes responsible for teaching the session. Otherwise, a FIPA Contract Net interaction protocol is launched to find another agent that can handle the event. If all agents are busy, they propose their closest sessions to the requested session. The initiating agent must then select the least busy one according to the timetables. The only restriction is that no agent can exceed its maximum working hours. The sequence diagram in Figure 4 illustrates this interaction.

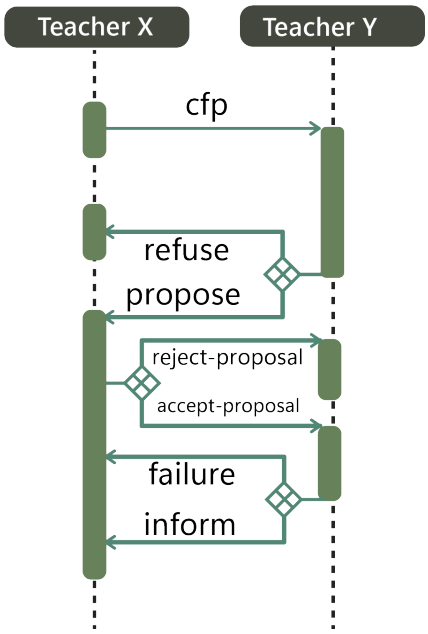


Figure 4. FIPA Contract Net interaction protocol—without organization.

Figure 5 presents a simplified code schema that handles events perception and, according to the event, determines whether to satisfy it or initiate negotiation.

```
class Perception extends TickerBehaviour{
   protected void onTick() {
      // retrieve an event
      if(isCompatibleEvent(event) && isNotExhausted()){
      // start a satisfaction strategy
      }
      else{
      // start a negotiation
      }
   }
}
```

Figure 5. The perception of an event by a teacher.

In the second example, we adopt an organizational perspective. The system is modeled, using the AGR organizational model, with two roles: teacher and department head. Teacher agents are members of at least one group (a department), and each department is headed by a department head agent. In this case, the event is characterized by the day, time, type of service, and department.

A department head agent examines only the events relevant to its department. If an event is captured and one of the teacher agents in that department can provide the requested service, the same protocol as in the non-organizational case is applied. If all the teacher agents respond negatively, the department head uses its authority to assign the event to a teacher agent. The sequence diagram in Figure 6 illustrates this interaction.

If none of the department's teacher agents can provide the required service, the department head sends a request to another department head who can satisfy the event. The latter, however, does not use its authority to satisfy the event. The sequence diagram in Figure 7 illustrates this interaction.

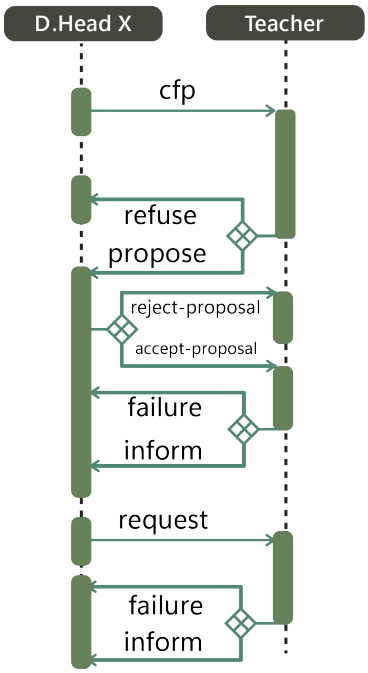


Figure 6. The interaction diagram for the scenario with organization (situation 01).

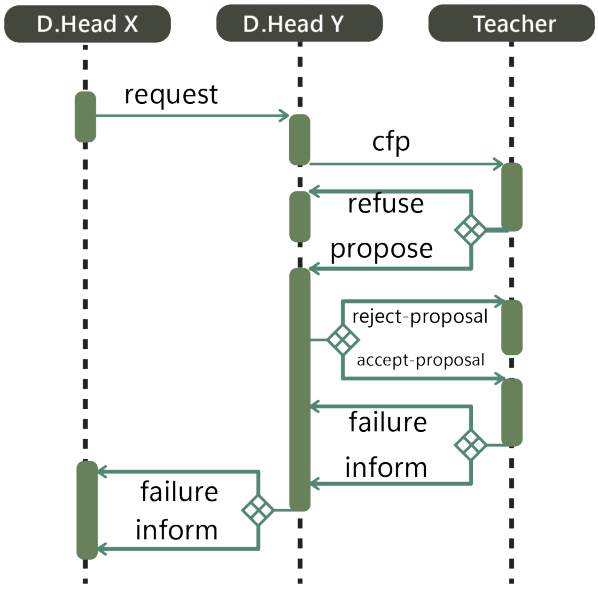


Figure 7. The interaction diagram for the scenario with organization (situation 02).

The two case studies are executed using almost the same scenario: four services represent the courses (ADT, Operating Systems, Networks, Algebra, and Analysis), while there are four teacher agents (*T1*, *T2*, *T3*, and *T4*). Table 1 presents the case study without organization. It shows, for each course, the number of required services (i.e., the number of sessions) and the agent qualified to deliver the course.

Table 1. Execution configuration of the case study without organization.

Courses (Services)	Number of Sessions	Courses for T1	Courses for T2	Courses for T3	Courses for T4
ADT	7	X			
Operating System	6	X	X		
Network	4	X	X		
Algebra	6			X	
Analysis	6				X

In the second scenario, the case study is implemented according to the AGR organizational model. Two groups are introduced to represent departments (the IT Department and the Mathematics Department). Each department is headed by a department head agent (Head1 for the IT Department and Head2 for the Mathematics Department). Also, agents T1 and T2 belong to the IT Department, while agents T3 and T4 belong to the Mathematics Department. Table 2 presents the number of sessions (services) required by each department. It is worth noting that the sum of the required sessions for each course is equal to the number of sessions specified in the previous scenario.

Table 2. Required services by department.

Courses (Services)	Required Services in Mathematics Department	Required Services in IT Department
ADT	3	4
Operating System	2	4
Network	1	3
Algebra	4	2
Analysis	4	2

After executing the first case study with the corresponding scenario, the events obtained are presented in Table 3. In addition, Figure 8 shows the results of the *Service Wealth (SW)* metric. Based on this observation, the *T1 agent* (shown as a red line) can be considered more autonomous than the others with respect to this metric. To illustrate how *SW* evolves over time, we report the system state at key timestamps. At 28,615 ms, the system creates agent *T1*, which provides three services; since the *T1 agent* is the sole agent at that time, *SW(T1)* = 3/3. At 35,354 ms, agent *T2* is created with two services, increasing the system total to five services and producing *SW(T1)* = 3/5 and *SW(T2)* = 2/5 (shown as a blue line). At 40,402 ms, agent *T3* is created with one service, yielding *SW(T1)* = *3/6*, *SW(T2)* = 2/6, and *SW(T3)* = 1/6 (shown as a green line). Finally, at 45,475 ms, agent *T4* is added with one service, so *SW(T1)* = 3/7, *SW(T2)* = 2/7, *SW(T3)* = 1/7, and *SW(T4)* = 1/7 (shown as a yellow line). Note that the *SW* of agent *T1* decreases gradually with the creation of new agents; this is quite normal because the creation of new agents means that additional services are available. Despite all of this, the value of *SW(T1)* remains the highest compared to the other agents.

Table 3. The relevant events in an unorganized scenario.

Without Organization	
Time /ms	**Events**
0	The beginning of execution
28,615	Creation of the agent T1
28,630	Behaviors scheduling of agent T1
35,354	Creation of agent T2
35,385	Behaviors scheduling of agent T2
40,402	Creation of agent T3
40,418	Behaviors scheduling of agent T3
45,475	Creation of agent T4
45,491	Behaviors scheduling of agent T4

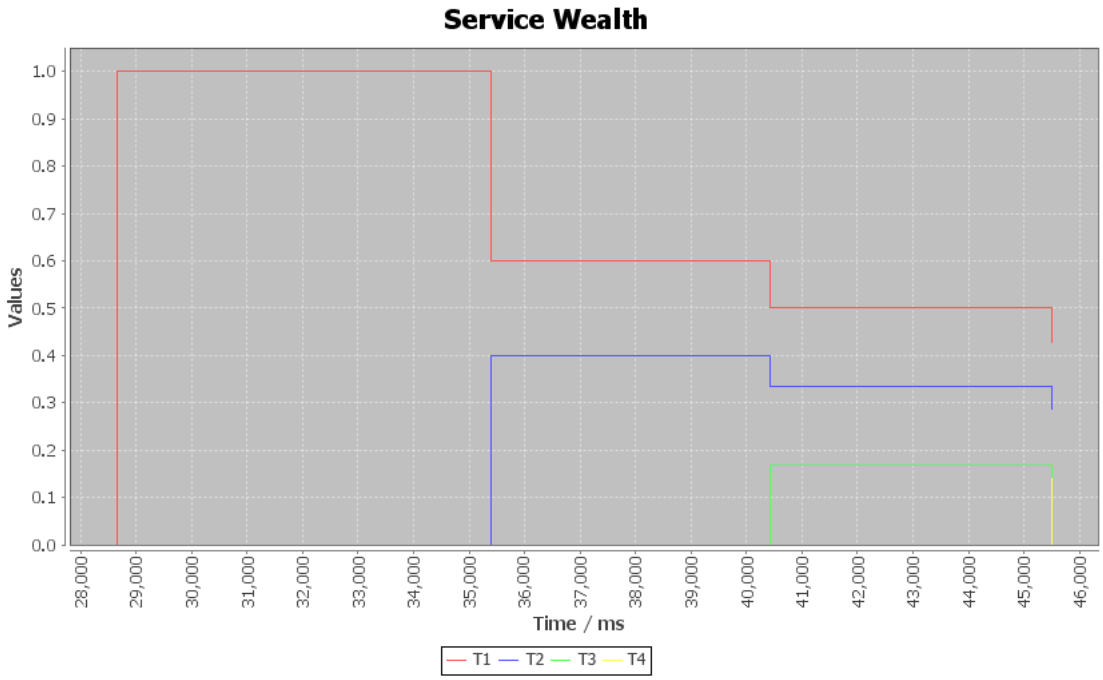


Figure 8. The results of the wealth of services. (The scenario without organization).

In addition to the previous observations, according to the *Frequency of Service Searches per Behavior (FoSSB)* metric, agent *T1* rarely needs to resort to service searches, as presented in Figure 9 (shown as a red line). In contrast, agents *T3* and *T4* possess the lowest *SW*, which explains their repeated reliance on service searches (shown, respectively, as green and yellow lines). Consequently, agent *T1* can be considered more autonomous than the others due to its broader range of skills.

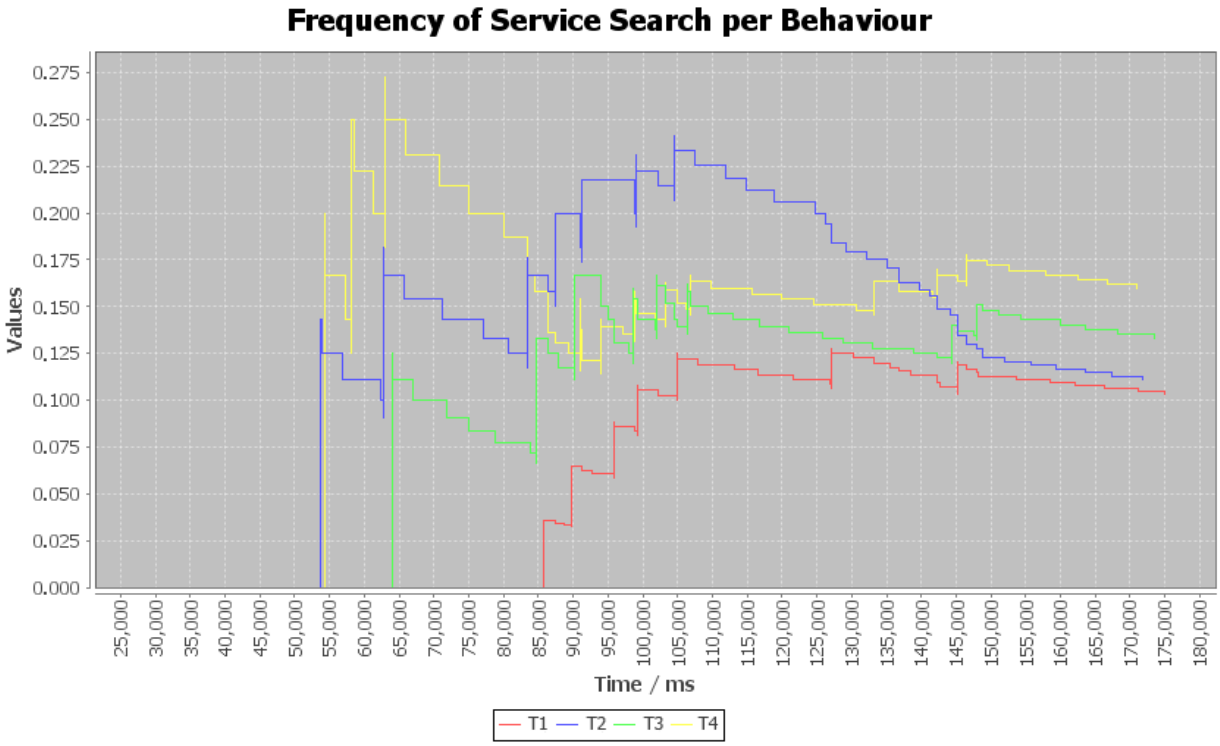


Figure 9. The results of service search frequency per behavior. (The scenario without organization).

Unlike the previous example, the organization scenario includes other behaviors such as registering teachers at the department level. Table 4 represents the relevant events in this scenario.

Table 4. The relevant events in a scenario with organization.

With Organization	
Time /ms	**Events**
0	The beginning of execution
42,669	Creation of agent ITDepH
42,700	Behaviors scheduling of agent ITDepH
51,863	Creation of agent MDepH
51,878	Behaviors scheduling of agent MDepH
68,578	Creation of agent T1
68,593	Behaviors scheduling of agent T1
76,743	Creation of agent T2
76,754	Behaviors scheduling of agent T2
86,520	Creation of agent T3
86,552	Behaviors scheduling of agent T3
94,303	Creation of agent T4
94,319	Behaviors scheduling of agent T4

This example demonstrates how the concept of the organization affects autonomy. The authority relationship between the department head and a teacher causes this phenomenon. Although the teacher responds negatively, the latter must obey the strict order of his department head, which reduces the teacher's autonomy. Therefore, we can say that a department head is more autonomous than a teacher.

For readability reasons, Figure 10 presents only the *Number of Provided Services per Demand (NoPSD)* metric of the *T1 agent*. With this foundation in place, the results of the *NoPSD* metric for agent *T1* reveal two distinct phases: an increasing phase followed by a decreasing phase.

In the first phase, the increase comes from the demands sent by the *ITDepH agent*, that *T1* was able to handle successfully. The ratio climbs steadily, showing that the *T1 agent* was keeping up with requests. The peak value of 2 is not surprising; it happens right when the *T1 agent* first registers with a department (one additional service that is not demanded), causing the metric to briefly rise above one before stabilizing.

In the second phase, the trend reverses. Here, many of the new demands are initiated by the *MDepH agent*, but the *T1 agent* is already overloaded with a busy timetable. Since the *T1 agent* cannot accept more services at this stage, the number of unanswered demands grows, pulling the *NoPSD* value down. In other words, new demands accumulate without being fulfilled, meaning the denominator of the metric grows faster than the numerator, leading to a decline in the *NoPSD* metric.

The authority relationship is not the only factor contributing to the autonomy of a department head compared to a teacher; it is also influenced by the number of behaviors of the department head. In our example, a teacher's declaration contains two behaviors (registration and waiting for messages). In contrast, a department head has an additional behavior—perception—bringing the total to five behaviors. Thus, the Be-

havioral Wealth of a teacher is *BW(Teacher)* = 2/5, while that of the department head is *BW(DepartmentHead)* = 3/5. Figure 11 shows the results for Behavioral Wealth.

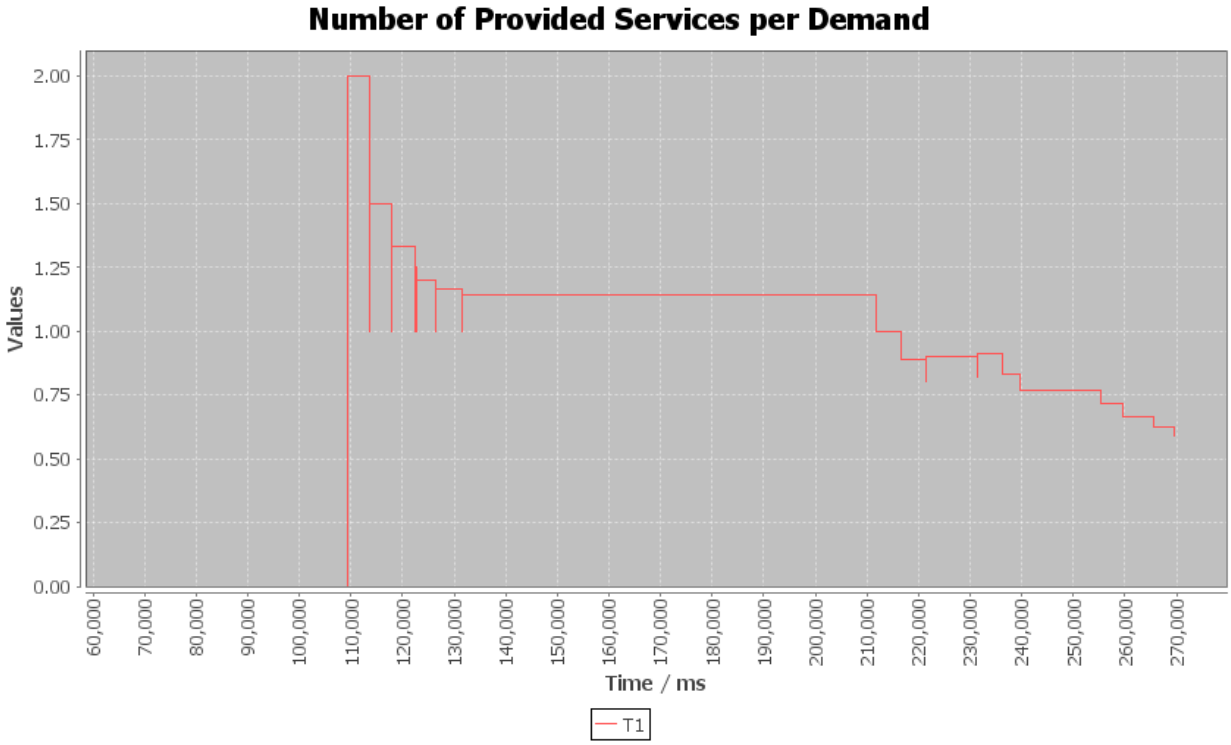


Figure 10. Results on the Number of Provided Services per Demand by agent T1 (with organization).

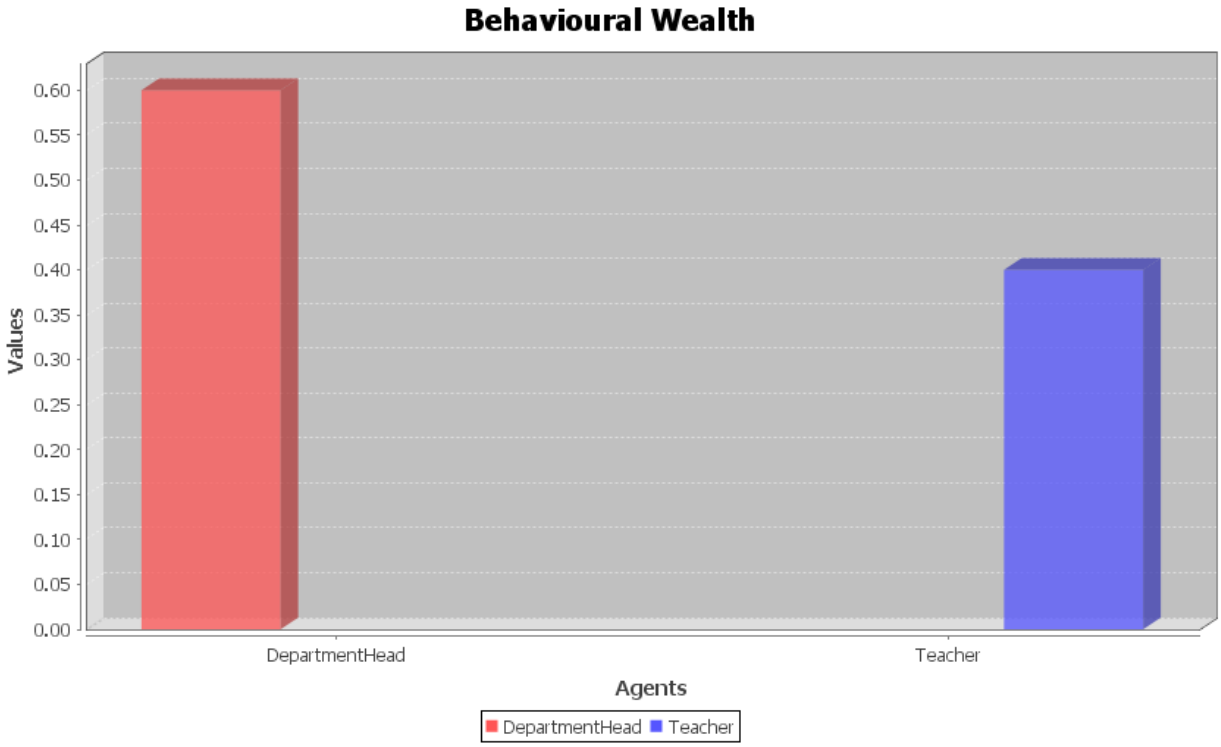


Figure 11. The results of Behavioral Wealth (in the scenario within an organization).

Moreover, we observe that the head of the *Information Technology* department can provide three types of service (ADT, Operating System, and Network), whereas the head of the *Mathematics* department offers only two (Algebra and Analysis). Consequently, the head

of the *Mathematics* department is less autonomous, as they resort to service searches more frequently than their counterpart in the *Information Technology* department. This conclusion is supported by the *Frequency of Service Searches per Behavior (FSSB)* metric illustrated in Figure 12.

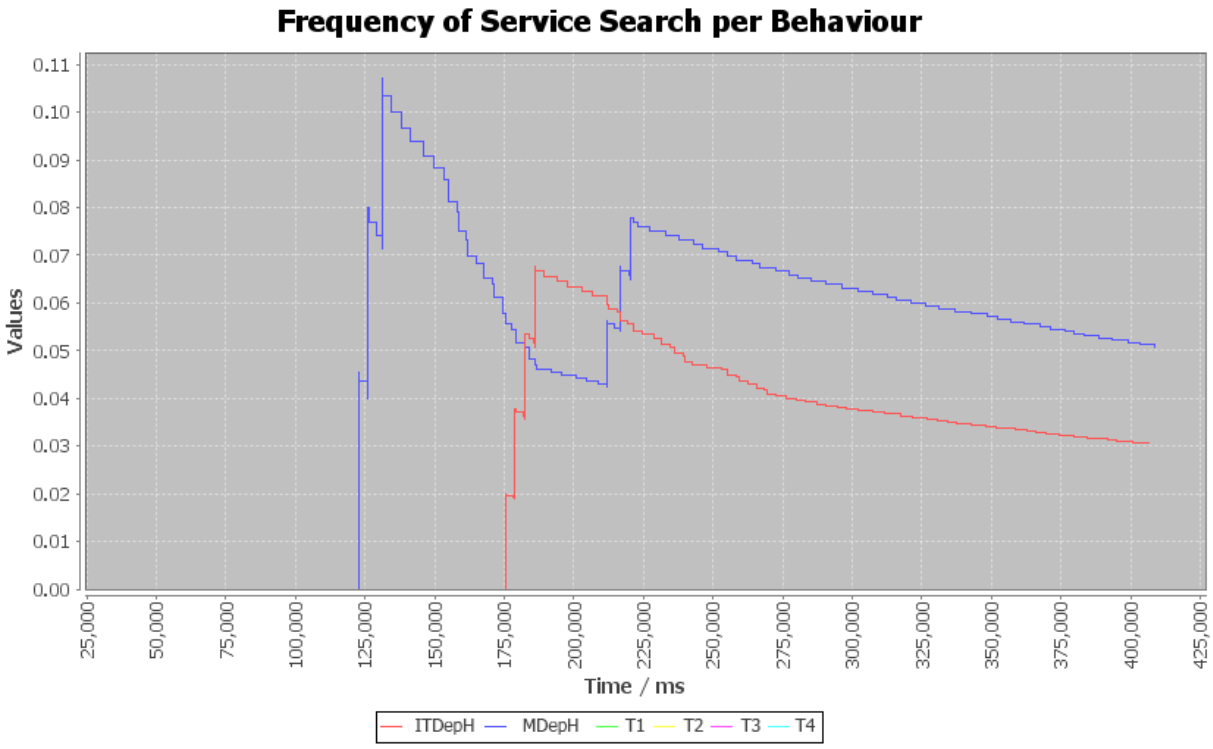


Figure 12. The results of service search frequency per behavior (in the scenario within an organization).

Moreover, it can be observed that the search process is restricted exclusively to department heads, despite the presence of teachers. This restriction primarily arises from the perception process, which applies only to department heads.

As a result, we observe that the organization impacts autonomy, particularly regarding the effectiveness of service requests due to authority relationships. Additionally, the organization reduces the need for service discovery because it enables agents to be aware of the various services implemented by each agent.

In the scenario without an organizational structure, agents are required to manage all interaction types, including CFP, ACCEPT-PROPOSAL, INFORM, PROPOSE, REFUSE, REJECT-PROPOSAL, and FAILURE. This significantly reduces their efficiency, as they become overloaded with communication tasks rather than concentrating on their primary execution roles. Moreover, the high volume of exchanged messages leads to system congestion, which in turn degrades overall performance.

With the organizational structure in place, the ITDepH and MDepH agents absorb a significant portion of the interactions, particularly messages such as CFP, INFORM, PROPOSE, REFUSE, and REQUEST. This is important because it reduces overall message density by relieving agents of communication responsibilities, thereby allowing them to concentrate on their primary execution roles. This effect is clearly illustrated in Figures 13–16.

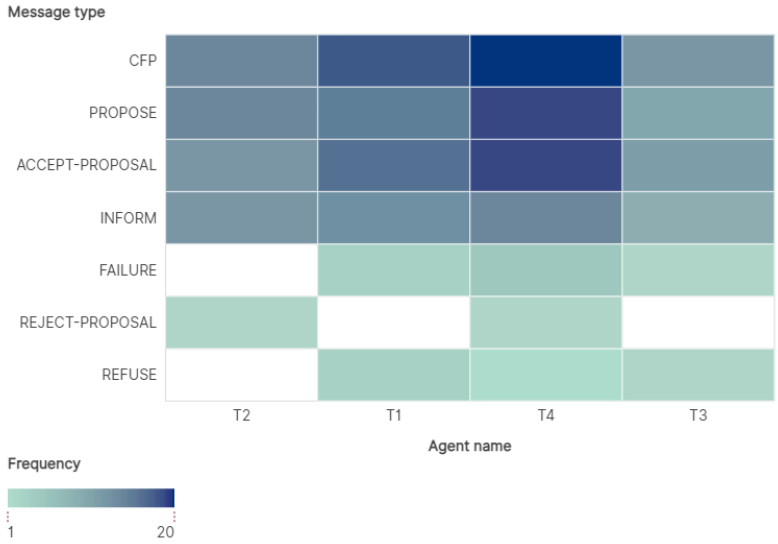


Figure 13. The density of messages (without organization).

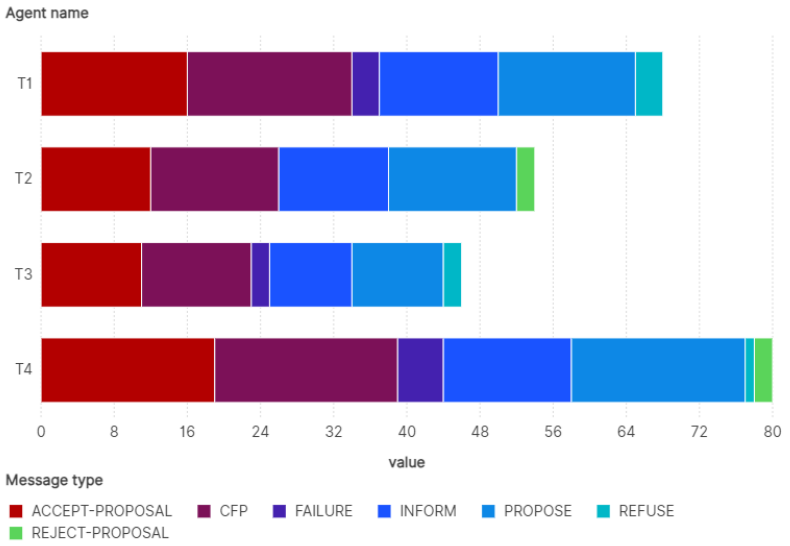


Figure 14. The number of messages per agent (without organization).



Figure 15. The density of messages (with organization).

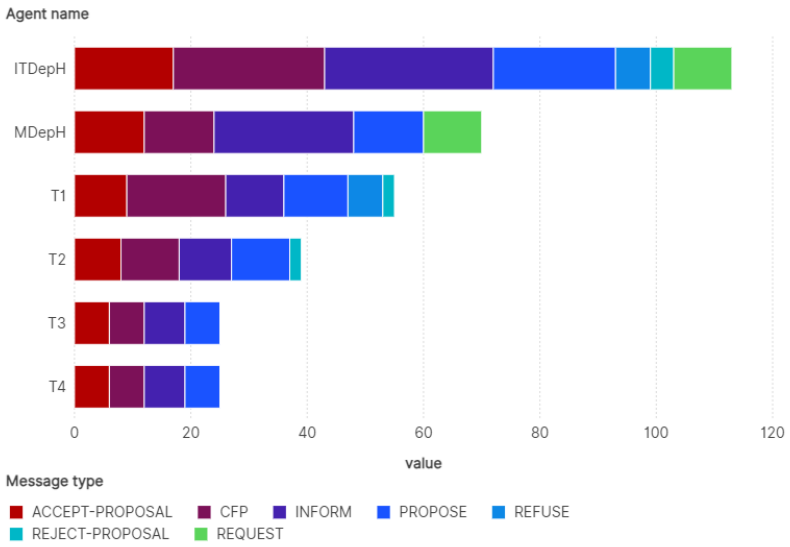


Figure 16. The number of messages per agent (with organization).

6. Discussion

Measurement is an important field in software engineering. Proposing metrics in this field provides several advantages, such as controlling the quality of the software, predicting the attributes of future software, or studying the relationships between software attributes. In particular, in agent-oriented software engineering, several attributes have been studied through the proposal of metrics to measure them.

Autonomy is one of the most important attributes of agents. Consequently, several studies have focused on analyzing it by proposing frameworks that highlight its facets or by introducing metrics to measure it [19–21]. In this context, although organization is theoretically known as a good method to manage the autonomy of agents without direct control, there is no empirical study that validates this idea. Therefore, our work aims to study the impact of organization on autonomy by proposing several metrics.

Our proposed approach, provides several advantages. First, the proposed metrics are dynamic; consequently, they are suitable for the dynamic nature of autonomy, especially in systems that use adjustable autonomy. Second, it is based on a minimalist specification of autonomy, namely the requirement of services. In fact, this minimalist specification makes our proposal extensible to support more sophisticated specifications. Furthermore, our work attempts to empirically analyze the relationship between two concepts: autonomy and organization. Consequently, the results can be used by developers to better manage the development of multi-agent systems (for example, by proposing the appropriate level of control over agents).

Despite the several advantages provided by this work, it suffers from several limitations that open the door to future improvement. First, autonomy, as presented previously, is a complex concept that can include motivation, constraints during decision-making, and available resources. However, we have based our work on a minimalist definition related to the requirement of services. In fact, in several multi-agent platforms (such as *JADE* [42]), the goal of the agent is not explicitly specified. Consequently, it becomes difficult to measure implicit attributes. Nevertheless, it is possible to propose additional metrics related to the goals of agents in multi-agent platforms that support these concepts (such as *JADEX* [43]). Moreover, this work is closely related to a specific organizational model (*AGR* [10]) and a specific multi-agent platform (*JADE* [42]). This choice is natural because the metrics operate at a low level of abstraction and are related to the execution of specific instructions. In this context, in future work, we propose to extend this approach to

support more facets of autonomy, more organizational models, and more multi-agent platforms. Consequently, a layered framework that allows passing from high-level abstraction to low-level abstraction seems a good approach to mastering this concept. In particular, a framework such as *GQM* [44] can be used to support such a proposition. Finally, although our system was executed with a limited number of agents, this is justified. On the one hand, the cognitive nature of the case study requires a limited number of agents. On the other hand, increasing the number of agents requires studying the concept of scalability and its relationship with the previous concepts (autonomy and organization).

7. Conclusions and Future Work

The multi-agent paradigm is a very potent paradigm that enables the design and development of complex systems. This software paradigm has introduced several new concepts, including autonomy, reactivity, proactivity, and organization. Autonomy is an intrinsic characteristic of agents. Thanks to this characteristic, agents can achieve their goals without external control. However, predicting full autonomy in practice is challenging. In this context, organization is closely related to autonomy because it allows controlling agents without direct intervention. The main objective of this work is to measure, in an objective manner, how organization affects autonomy. To validate our claims, we proposed several metrics to evaluate agent autonomy and applied them to a case study implemented in two versions: one with organization and one without. The proposed metrics are applied thanks to a tool we developed for this purpose using aspect-oriented programming. As a result, we can conclude that the organization can limit the autonomy of agents in certain dimensions due to authority relationships and the search for services. However, it enhances the effectiveness of agents by reducing the communication load and allowing agents to focus on their primary roles.

As future work, we propose exploring the relationship between autonomy and other related agent characteristics, such as proactivity. In addition, we plan to apply the proposed metrics to other organizational models.

Author Contributions: Conceptualization, Z.T., T.M. and F.M.; methodology, Z.T. and D.N.; software, Z.T.; validation, V.G., T.M. and F.M.; resources, D.N.; writing—original draft preparation, Z.T., T.M., D.N., V.G. and F.M.; writing—review and editing, Z.T., D.N., T.M., V.G. and F.M.; visualization, Z.T.; supervision, T.M.; project administration, F.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The complete source code for this project, including both the implementation without organization and with organization, as well as the implemented metrics, is publicly available on GitHub at https://github.com/getset04/MAS-Organization-Autonomy, (accessed on 24 September 2025).

Acknowledgments: This research work was supported by the General Direction of Scientific Research and Technological Development (DGRSDT) of the Algerian Higher Education and Scientific Research Ministry. We would like to thank the DGRSDT for its support in the achievement of this work.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AGR	Agent–Grou–Role
MAS	Multi-Agent Systems
BW	Behavioral Wealth
SW	Service Wealth
FoSST	Frequency of Service Searches per Time
FoSSB	Frequency of Service Searches per Behavior
NoSS	Number of Service Search
NoSDB	Number of Service Demands per Behavior
NoPSD	Number of Provided Services per Demand
JADE	Java Agent DEvelopment Framework
JADEX	Java Agent DEvelopment Framework Extension
GQM	Goal Question Metric

References

1. Wooldridge, M. *An Introduction to Multiagent Systems*; John Wiley & Sons:Hoboken, NJ, USA, 2009.
2. Dorri, A.; Kanhere, S.S.; Jurdak, R. Multi-agent systems: A survey. *IEEE Access* **2018**, *6*, 28573–28593. [CrossRef]
3. Fisher, M.; Mascardi, V.; Rozier, K.Y.; Schlingloff, B.H.; Winikoff, M.; Yorke-Smith, N. Towards a framework for certification of reliable autonomous systems. *Auton. Agents-Multi-Agent Syst.* **2021**, *35*, 8. [CrossRef]
4. Cortés, J.; Egerstedt, M. Coordinated control of multi-robot systems: A survey. *SICE J. Control. Meas. Syst. Integr.* **2017**, *10*, 495–503. [CrossRef]
5. Mostafa, S.A.; Ahmad, M.S.; Mustapha, A. Adjustable autonomy: A systematic literature review. *Artif. Intell. Rev.* **2019**, *51*, 149–186. [CrossRef]
6. Li, D.; Ge, S.S.; He, W.; Ma, G.; Xie, L. Multilayer formation control of multi-agent systems. *Automatica* **2019**, *109*, 108558. [CrossRef]
7. Chen, F.; Ren, W. *On the Control of Multi-Agent Systems: A Survey*; Foundations and Trends® in Systems and Control: Hanover, MA, USA, 2019; Volume 6, pp. 339–499.
8. Tranier, J. Vers une Vision Intégrale des Systèmes Multi-Agents: Contribution à l'intégration des Concepts d'Agent, d'Environnement, d'Organisation et d'Institution. Ph.D. Thesis, Université Montpellier II-Sciences et Techniques du Languedoc, Montpellier, France, 2007.
9. Braynov, S.; Hexmoor, H. Quantifying relative autonomy in multiagent interaction. In *Agent Autonomy*; Springer: Boston, MA, USA, 2003; pp. 55–73.
10. Ferber, J.; Gutknecht, O.; Michel, F. From agents to organizations: An organizational view of multi-agent systems. In Proceedings of the International Workshop on Agent-Oriented Software Engineering, Melbourne, VIC, Australia, 15 July 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 214–230.
11. Ferber, J.; Michel, F.; Báez, J. AGRE: Integrating environments with organizations. In Proceedings of the International Workshop on Environments for Multi-Agent Systems, New York, NY, USA, 19 July 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 48–56.
12. Mansour, S.; Ferber, J. Agent Groupe Rôle et Service: Un modèle organisationnel pour les systèmes multi-agents ouverts. In Proceedings of the Actes du Colloque JFSMA, Carcassonne, France, 17–19 October 2007.
13. Souidi, M.E.H.; Songhao, P.; Guo, L.; Lin, C. Multi-agent cooperation pursuit based on an extension of AALAADIN organisational model. *J. Exp. Theor. Artif. Intell.* **2016**, *28*, 1075–1088. [CrossRef]
14. Chebout, M.S.; Marir, T.; Gupta, V. NorAGR: A NORmative Agent Groupe Role Model for Organizational Centered Open Multi-Agent Systems. In Proceedings of the 2023 Third International Conference on Theoretical and Applicative Aspects of Computer Science (ICTAACS), Skikda, Algeria, 5–6 December 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 1–5.
15. Abbas, H.A.; Shaheen, S. Towards a hybrid MAS organizational model: Combining the ACMAS and OCMAS viewpoints. *Int. J. Organ. Collect. Intell. (IJOCI)* **2017**, *7*, 18–50. [CrossRef]
16. Boudjidj, A.; Merah, E.; Souidi, M.E.H. Towards a formal multi-agent organizational modeling framework based on category theory. *Informatica* **2021**, *45*, 277–288. [CrossRef]
17. Laouadi, M.A.; Mokhati, F.; Seridi-Bouchelaghem, H. A formal framework for organization-centered multi-agent system specification: A rewriting logic based approach. *Multiagent Grid Syst.* **2017**, *13*, 395–419. [CrossRef]

18. Sudeikat, J.; Kohler-Bußmeier, M. Towards integrating multi-agent organizations in OPC UA for developing adaptive cyber-physical systems. In Proceedings of the INFORMATIK 2022, Hambrug, Germany, 26–30 September 2022.
19. Luck, M.; d'Inverno, M. A Formal Framework for Agency and Autonomy. In Proceedings of the ICMAS, San Francisco, CA, USA, 12–14 June 1995; Volume 95, pp. 254–260.
20. Barber, K.S.; Martin, C.E. Agent autonomy: Specification, measurement, and dynamic adjustment. In Proceedings of the Theautonomy Control Software Workshop at Autonomous Agents, Seattle, WA, USA, 1–5 May 1999 ; Citeseer: Princeton, NJ, USA, 1999; Volume 1999, pp. 8–15.
21. Alonso, F.; Fuertes, J.L.; Martínez, L.; Soza, H. Towards a set of measures for evaluating software agent autonomy. In Proceedings of the 2009 Eighth Mexican International Conference on Artificial Intelligence, Guanajuato, Mexico, 9–13 November 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 73–78.
22. Suzanne Barber, K.; Goel, A.; Martin, C.E. Dynamic adaptive autonomy in multi-agent systems. *J. Exp. Theor. Artif. Intell.* **2000**, *12*, 129–147. [CrossRef]
23. Scerri, P.; Pynadath, D.; Tambe, M. Adjustable autonomy in real-world multi-agent environments. In Proceedings of the Fifth International Conference on Autonomous Agents, Montreal, QB, Canada, 28 May–1 June 2001; pp. 300–307.
24. Goodrich, M.A.; Adams, J.A.; Scheutz, M. Autonomy reconsidered: Towards developing multi-agent systems. In *Intelligent Systems and Applications, Proceedings of the 2021 Intelligent Systems Conference (IntelliSys), Amsterdam, The Netherlands, 3–4 September 2022*; Springer: Berline/Heidelberg, Germany, 2022; pp. 573–592.
25. Dodig-Crnkovic, G.; Burgin, M. A systematic approach to autonomous agents. *Philosophies* **2024**, *9*, 44. [CrossRef]
26. Zhou, W.; Jiang, Y.E.; Li, L.; Wu, J.; Wang, T.; Qiu, S.; Zhang, J.; Chen, J.; Wu, R.; Wang, S.; et al. Agents: An open-source framework for autonomous language agents. *arXiv* **2023**, arXiv:2309.07870. [CrossRef]
27. Putta, P.; Mills, E.; Garg, N.; Motwani, S.; Finn, C.; Garg, D.; Rafailov, R. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv* **2024**, arXiv:2408.07199. [CrossRef]
28. Ferrag, M.A.; Tihanyi, N.; Debbah, M. From llm reasoning to autonomous ai agents: A comprehensive review. *arXiv* **2025**, arXiv:2504.19678. [CrossRef]
29. Abbas, H.A.; Shaheen, S.I.; Amin, M.H. Organization of multi-agent systems: An overview. *arXiv* **2015**, arXiv:1506.09032. [CrossRef]
30. Dignum, V.; Noriega, P.; Sensoy, M.; Sichman, J.S. *Coordination, Organizations, Institutions, and Norms in Agent Systems XI*; Springer: Cham, Switzerland, 2016.
31. Boes, J.; Migeon, F. Self-organizing multi-agent systems for the control of complex systems. *J. Syst. Softw.* **2017**, *134*, 12–28. [CrossRef]
32. Lhaksmana, K.M.; Murakami, Y.; Ishida, T. Role-based modeling for designing agent behavior in self-organizing multi-agent systems. *Int. J. Softw. Eng. Knowl. Eng.* **2018**, *28*, 79–96. [CrossRef]
33. Wang, T.; Dong, H.; Lesser, V.; Zhang, C. Roma: Multi-agent reinforcement learning with emergent roles. *arXiv* **2020**, arXiv:2003.08039. [CrossRef]
34. Pacheco, O. Autonomy in an organizational context. In Proceedings of the International Workshop on Computational Autonomy, Melbourne, VIC, Australia, 14 July 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 195–208.
35. Schillo, M.; Fischer, K. A taxonomy of autonomy in multiagent organisation. In Proceedings of the International Workshop on Computational Autonomy, Melbourne, VIC, Australia, 14 July 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 68–82.
36. Mahmoud, M.A.; Ahmad, M.S.; Mohd Yusoff, M.Z.; Mustapha, A. A review of norms and normative multiagent systems. *Sci. World J.* **2014**, *2014*, 684587. [CrossRef] [PubMed]
37. Carabelea, C.; Boissier, O.; Florea, A. Autonomy in multi-agent systems: A classification attempt. In Proceedings of the International Workshop on Computational Autonomy, Melbourne, VIC, Australia, 14 July 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 103–113.
38. Beavers, G.; Hexmoor, H. Types and limits of agent autonomy. In Proceedings of the International Workshop on Computational Autonomy, Melbourne, VIC, Australia, 14 July 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 95–102.
39. Marir, T.; Mokhati, F.; Bouchelaghem-Seridi, H.; Benaissa, B. Dynamic Metrics for Multi-agent Systems Using Aspect-Oriented Programming: Application to DIMA Platform. In Proceedings of the German Conference on Multiagent System Technologies, Klagenfurt, Austria, 27–30 September 2016; Springer: Cham, Switzerland, 2016; pp. 58–72.
40. Kiczales, G.; Lamping, J.; Mendhekar, A.; Maeda, C.; Lopes, C.; Loingtier, J.M.; Irwin, J. Aspect-oriented programming. In Proceedings of the European Conference on Object-Oriented Programming, Jyväskylä, Finland, 9–13 June 1997; Springer: Berline/Heidelberg, Germany, 1997; pp. 220–242.
41. Kiczales, G.; Hilsdale, E.; Hugunin, J.; Kersten, M.; Palm, J.; Griswold, W.G. An Overview of AspectJ. In *ECOOP 2001—Object-Oriented Programming, Proceedings of the 15th European Conference, Budapest, Hungary, 18–22 June 2001*; Lecture Notes in Computer Science; Knudsen, J.L., Ed.; Springer: Berline/Heidelberg, Germany, 2001; Volume 2072, pp. 327–353. [CrossRef]
42. JADE. Java Agent DEvelopment Framework (JADE). 2024. Available online: https://jade.tilab.com/ (accessed on 21 June 2024).

43. Pokahr, A.; Braubach, L.; Lamersdorf, W. Jadex: A BDI reasoning engine. In *Multi-Agent Programming: Languages, Platforms and Applications*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 149–174.
44. Van Solingen, R.; Basili, V.; Caldiera, G.; Rombach, H.D. Goal question metric (gqm) approach. In *Encyclopedia of Software Engineering*; John Wiley & Sons: Hoboken, NJ, USA, 2002.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.