information

Review

Overview of Software Agent Platforms Available in 2023

Zofia Wrona [1], Wojciech Buchwald [1], Maria Ganzha [1], Marcin Paprzycki [2,*], Florin Leon [3], Noman Noor [1] and Constantin-Valentin Pal [3]

1 Faculty of Mathematics and Information Science, Warsaw University of Technology, 00-662 Warsaw, Poland; zofia.wrona.stud@pw.edu.pl (Z.W.); wojciech.buchwald.stud@pw.edu.pl (W.B.); Maria.Ganzha@pw.edu.pl (M.G.); noman.noor.stud@pw.edu.pl (N.N.)
2 Systems Research Institute, Polish Academy of Sciences, 01-447 Warsaw, Poland
3 Faculty of Automatic Control and Computer Engineering, "Gheorghe Asachi" Technical University of Iași, 700050 Iași, Romania; florin.leon@academic.tuiasi.ro (F.L.); constantin-valentin.pal@student.tuiasi.ro (C.-V.P.)
* Correspondence: marcin.paprzycki@ibspan.waw.pl

Abstract: Agent-based computing remains an active field of research with the goal of building (semi-)autonomous software for dynamic ecosystems. Today, this task should be realized using dedicated, specialized frameworks. Over almost 40 years, multiple agent platforms have been developed. While many of them have been "abandoned", others remain active, and new ones are constantly being released. This contribution presents a historical perspective on the domain and an up-to-date review of the existing agent platforms. It aims to serve as a reference point for anyone interested in developing agent systems. Therefore, the main characteristics of the included agent platforms are summarized, and selected links to projects where they have been used are provided. Furthermore, the described platforms are divided into general-purpose platforms and those targeting specific application domains. The focus of the contribution is on platforms that can be judged as being under active development. Information about "historical platforms" and platforms with an unclear status is included in a dedicated website accompanying this work.

Keywords: agent systems; agent-based systems; multi-agent systems; agent platforms; modeling; simulation; swarm intelligence; artificial intelligence

Citation: Wrona, Z.; Buchwald, W.; Ganzha, M.; Paprzycki, M.; Leon, F.; Noor, N.; Pal, C.-V. Overview of Software Agent Platforms Available in 2023. *Information* **2023**, *14*, 348. https://doi.org/10.3390/info14060348

Academic Editor: Katsuhide Fujita

Received: 4 June 2023
Revised: 11 June 2023
Accepted: 12 June 2023
Published: 18 June 2023

1. Introduction

To the best of our knowledge, it has been over 10 years since the last comprehensive overview of tools available for the development of agent systems was published [1]. Since agent systems remain an active research area and software agents are starting to be applied in new domains [2], it can be suggested that it is the right time to reflect on the current state of the art regarding the platforms that can be used for their development. To provide background for non-specialists, this contribution begins by briefly outlining key historical developments in the area of software agents and agent systems. However, it is important to stress that this work is not an introduction to the field of (multi-)agent systems. The readers interested in the topic are invited to study general books on the subject, such as [3,4]. Before proceeding, let us also note that this work is an update to the technical report published in arXiv in July 2020 [5]. In this regard, comparing the content of the two reports prepared three years apart on the same topic may also be valuable for capturing the dynamics of the changes.

1.1. Short History of Agent Systems

Perhaps one of the pivotal moments in the development of agent systems was the Workshop on Distributed Artificial Intelligence held at MIT in June 1980, where 22 researchers presented their results and ideas [6]. After that, the 1980s saw the first attempts at defining the main concepts of the agents' domain. Interestingly, among these, one can also recognize the key problems that are still being studied today.

In 1984, Axelrod showed how cooperation can emerge from the interaction between selfish entities without centralized control [7]. He discussed several strategies for the iterated prisoner's dilemma, a game-theoretical problem that remains a topic of interest in political and social studies, as well as evolutionary biology.

In 1985, the actor model was introduced by Agha and Hewitt [8]. With all the "generalization" involved in this statement, it can be suggested that an actor can be seen as a "simplified version" of an agent. The actor is reactive as it only responds to the received messages, but it can be used to simulate more advanced behaviors, such as perceiving the environment and performing actions, in response to the incoming communication. The research involving actors contributed to the development of agent systems while also resulting in the development of its own tools, libraries, and platforms (e.g., Erlang [9], Akka [10], and Proto.Actor [11]). However, this pathway of research and development is beyond the scope of this work.

In 1986, Brooks elaborated on his rejection of the logical, symbolic approach to intelligence prevalent at that time and proposed the subsumption architecture [12]. Moreover, he implemented four robots capable of seemingly intelligent behavior without any symbolic internal representation of the environment. At the beginning of the 1990s, Maes worked together with Brooks at MIT to design robots for building a base on the Moon [13]. Afterwards, she focused on software agents for personalized information filtering [14] that were later abstracted into what is now commonly viewed as an agent—an autonomous entity [15,16].

Furthermore, in 1986, the first (micro-)simulations were conceived; e.g., in the pursuit domain, where several predators aim at encircling and/or capturing a prey [17]. Nowadays, simulations in different domains are one of the highly popular application areas for multi-agent systems.

Based on psychological studies of practical reasoning ([18]), in 1987, Georgeff and Lansky developed the so-called procedural reasoning system (PRS), established on the foundation of the belief–desire–intention (BDI) model for intelligent agents [19]. A formalization of the BDI architecture, which remains very popular in the agent domain, can be found in [20]. It explicitly includes the agent's beliefs about the state of the environment and its own state. It also incorporates the concepts of establishing the means for reaching a specific goal and for creating a plan.

The first agents (agent systems) were implemented using the means available at the time; e.g., in Lisp [21] or Prolog [22]. However, as the complexity of applications increased and more experience was gathered, the need for specialized frameworks dedicated to constructing agents was recognized. One of the first attempts at delivering a complete agent platform was AGENT0 [23], a language that incorporated the idea of agent-oriented programming [24] where the agent has complete control over its own state (beliefs, capabilities) and behaviors (responding to messages, commitments).

The 1990s witnessed a steady stream of development in the agent systems domain where separate areas of interest can be identified. During that time, significant advances materialized in the theoretical study of agent systems. For instance, the ARCHON project [25] proposed a general-purpose architecture that could facilitate cooperative problem-solving in industrial applications.

The knowledge query and manipulation language (KQML) was formulated within the DARPA knowledge sharing effort [26] and proposed as a standardized means of agent communication. It separated the intent of the message, in the form of a so-called "performative" (inspired by speech act theory [27]), from its actual content. That effort was further refined within the FIPA agent communication language (ACL) [28] definition. Separately, a specific agent-based language, AgentSpeak, inspired by the PRS was formally introduced in [29]. Moreover, software agents were described and discussed in core contributions by Wooldridge and Jennings [30–32].

Related interests emerged in the field of autonomous cars, which have been conceptualized as autonomous agents. Here, several agent architectures were proposed (e.g., Touring Machine [33], InteRRaP [34]) and the first self-driving vehicles were put to the test (e.g., ALVINN [35] and, later, the Nomad rover [36] and Stanley [37]).

Another area of multi-agent systems' application that expanded greatly during that time was related to social simulations. Here, a few notable examples are the Sugarscape emergent economic model [38], the evolution of social corruption [39], the study of the population dynamics of Cyber-Anasazi [40], and a model for civil violence leading to "artificial genocide" [41].

Reflecting on the connection to the industry, one should note the (multi-stage) European AgentLink project [42–44], which studied the application of agent systems in domains such as telecommunications, information management, electronic commerce, and manufacturing. It is easy to notice that the vision of software agents formulated within these activities is materializing now, among other places, in the world of the Internet of Things (IoT) [2].

An interesting European initiative from 2013 was the COST agreement technologies action [45]. It aimed at coordinating efforts related to the new paradigm for next-generation distributed systems based on the concept of agreement between computational agents. In such systems, autonomous software agents negotiate with each other, typically on humans' behalf, to reach mutually acceptable agreements.

Despite constant research activities, at the beginning of the new millennium, interest in agent systems gradually declined. However, it appears that, in the past few years, the field has been experiencing a revival of interest accommodated by increased awareness of the importance of agent systems. This interest especially concerns domains such as autonomous cars and drones (e.g., pick-up and delivery routing problem [46]), simulations (e.g., evacuation behavior during emergencies [47]), smart cities [48,49], and the IoT. In these areas, new technological advances in the communication infrastructure (e.g., 5G [50,51]) are facilitating further automation of and interconnections between intelligent devices.

Today, agent systems are extensively used in medical and social simulations, particularly those that examine the spread of infectious diseases (e.g., the impact of mobility restrictions on the spread of COVID-19 [52] and strategies for minimizing the spread of epidemics in populations [53]). Moreover, agent systems are being employed for modeling in the electromobility sector [54–57], as well as in electricity markets [58]. In addition, the agent-based approach is also being explored in the context of multi-agent deep reinforcement learning, which is useful in modeling realistic and complex environments (e.g., maintenance in manufacturing systems [59]).

Nevertheless, as far as industrial-grade systems are concerned, agent-based solutions are frequently being replaced with alternative approaches, such as microservices, serverless systems (FaaS), and workflow management systems (WMSs). This shift is driven by factors such as the broader range of technological options offered by agent system alternatives. For instance, microservices enable the implementation of various application services using different technologies tailored to their specific requirements. As such, they provide greater flexibility and adaptability within the overall system architecture. Furthermore, in many cases, the complexity of agent systems surpasses the actual needs and requirements of the tasks that are to be handled by the system, thereby increasing the cost of the system's development, maintenance (e.g., cross-compatibility), and deployment.

Regardless of the relative lack of popularity of agent systems in commercial applications, over time, the idea of their applicability has materialized in a broad range of different domains. Let us identify the most prominent ones (approaching them from the point of view of existing research contributions).

1.2. Applications of Agent Systems

In what follows, studies outlining particular use cases in each of the major fields that have profited from the employment of agent systems are provided. Note that this list is not comprehensive and that the references supplied here should be treated as mere examples. Readers are encouraged to look for more on their own if interested.

- *Social simulations*: Various scenarios regarding human interactions and the behaviors of whole communities [60–62]; academia simulations, including e-learning [63] and cooperative learning [64];
- *Mobility simulations*: Traffic situations, such as the avoidance of traffic jams, light control, and route choice [65,66]; ground transportation, mobility planning systems, and urban planning based on accessibility studies with dynamic populations [67]; microscopic pedestrian crowds [68]; and mapping passenger flow for market improvement, evacuation of buildings, and flight or air-traffic control in aviation [69];
- *Physical entities*: Robots and self-driving vehicles (cars, drones) seen as agents [70–72];
- *Environment and ecosystems*: Simulations in ecology [73,74] and biology, climate models, human and nature interaction (sometimes using geographic information systems), and epidemiology (spread of infections or diseases) [75,76];
- *Organizational simulations*: Planning and scheduling, enterprise and organizational behavior, workflow simulations [77], and implementing human–agent teams in corporate environments [78];
- *Economic studies*: Business, marketing, and economics (e.g., price forecasting in real-world markets [79–81]);
- *Medical applications*: Personalized healthcare and hospital management [82,83];
- *Industrial simulations*: Manufacturing and production, including with the use of holons [84,85];
- *Military applications*: Military combat simulations and air-defense scenarios [86–88];
- *Entertainment*: Rendering large-scale battles in movies [89], the video games industry [90,91], and motion capturing [92];
- *Distributed computing*: Allocating and negotiating resources in cloud computing [93,94] and communication in weakly connected clusters [95];
- *Coordination systems*: Coordination of hardware [96] and software [97–99] agents.

Considering the diversity of the areas of application of agent-based systems, it should be noted that different research problems can be associated with different individual requirements. Such requirements place demands on the developed system to incorporate specific features or adhere to a particular architectural approach. Those needs must be addressed by the underlying agent platforms, restricting them in some cases to a specific application domain and hence contributing to the diversity in the field. Therefore, in order to conduct a comprehensive review, it is necessary to first establish the means used in the selection and classification of agent platforms. Let us briefly outline the approach that was adopted in this contribution.

2. Structure of the Review

In light of the (1) the breadth and depth of applications of software agents, (2) the continuous development and growing maturity of existing agent platforms, and (3) the creation of new platforms, it was decided that an up-to-date overview of the state of the art may be of value. However, with the decrease in interest in agent systems (after 2005), many agent platforms have been abandoned. As a result, it is essential to summarize what is actually available today for those interested in implementing and experimenting with agent systems. In this context, the initial version of this contribution [5] was published in July 2020. The current version of the text takes into account the changes that have taken place during the last three years, as well as the results of further searching for agent platforms. However, when reflecting on the content of the earlier (2020) contribution, it was decided that information about obsolete agent platforms took up too much space (even if it was only presented as summary tables). Moreover, a number of platforms that still remain "usable" seem to be on a fast track to their demise. Therefore, a dedicated website (https://www.ibspan.waw.pl/~paprzyck/mp/cvr/research/agent_platforms_site/agent_platforms.html (accessed on 3 June 2023)) that complements this text has been developed. There, among other resources, "complete" information about "historical" platforms can be found. It also includes extensive information about platforms for which the future status is unclear (information on which has been omitted here). Furthermore, on the website, interested users may also find information about actor-based platforms that are outside of the scope of this study. In this context, it should be stressed that readers who would like to contribute (e.g., correct existing or add missing information) to the site are more than welcome to do so and should communicate with the contact author of this work.

Given the above factors, this contribution focuses on agent platforms, both free and commercial, that remain active. These platforms were divided, according to their intended use, into general-purpose ones and those that are domain-specific. Due to the vast number of general-purpose platforms, this group was further divided, based on the licensing, into open-source and commercial platforms. Furthermore, it was noticed that, among the open-source platforms, there are many that are exclusively dedicated to modeling and simulation. Hence, this group was also distinguished with a separate section. The domain-specific platforms were categorized according to the particular research areas in which they are applicable. The detailed platform classification is presented in Figure 1.

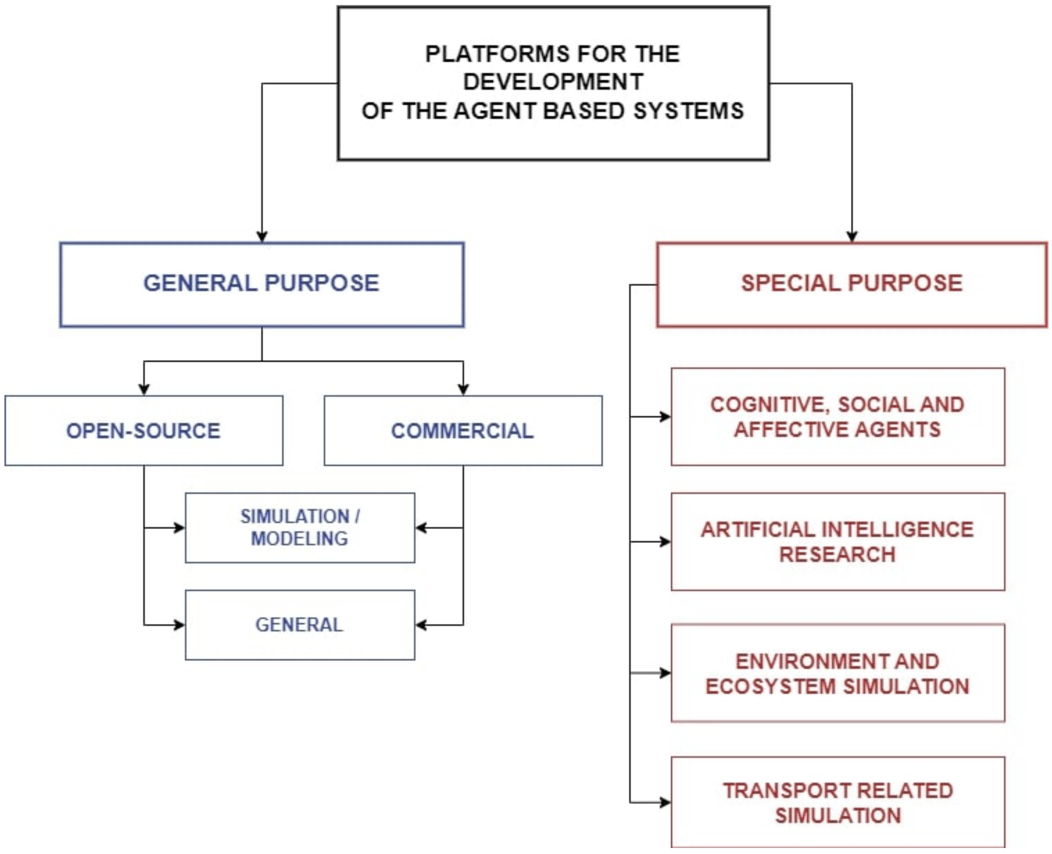
PLATFORMS FOR THE DEVELOPMENT OF THE AGENT BASED SYSTEMS
GENERAL PURPOSE
SPECIAL PURPOSE
OPEN-SOURCE
COMMERCIAL
SIMULATION / MODELING
GENERAL
COGNITIVE, SOCIAL AND AFFECTIVE AGENTS
ARTIFICIAL INTELLIGENCE RESEARCH
ENVIRONMENT AND ECOSYSTEM SIMULATION
TRANSPORT RELATED SIMULATION

Figure 1. Top-level categorization of platforms considered in this review.

Each group of platforms is summarized in the following manner. They are listed in tables that outline their most important features, including information regarding platform licensing, the programming language, the latest version, and a brief description. Additionally, the platforms are supplied with links to related websites (All websites have been accessed on 3 June 2023), such as the platform source code (if accessible) or platform homepage. For each link, wherever possible, an archived version is also included in case the website is no longer available (for links that could not be archived, screenshots are provided in the website accompanying this work). Additionally, the key features for each of the platforms are outlined in greater detail.

The scope of this contribution covers a wide range of agent platforms. Hence, it extends to the implementation of agent systems across various research domains while also considering frameworks that accommodate modern technological demands. In order to contextualize this work, let us examine other approaches adopted in the literature related to the agent platforms domain.

3. Related Work: Earlier Reviews of Agent Platforms

As the field of agent systems has matured, a number of authors have reviewed the "landscape of agent platforms" [100]. Some reviews are confined to specific application domains, such as traffic simulations [101,102], marketing [103], UAVs [104], networking [105], or land use [106]. Others, such as [107], assess the existing (at the time of writing) platforms from the perspective of the system structure and the supported methodology. There are also reviews, such as [108], that focus mainly on the platforms that can be used for educational and teaching purposes.

In some work, the emphasis is primarily put on the agent software itself, discussing its definitions, applications, features, and challenges. As an example, the survey in [109] provides a detailed overview of the diverse characteristics of agent systems; however, it does not include a comprehensive listing of platforms used in their development. Some of the older reviews, written in a similar manner, can be found in [110,111].

On the other hand, studies such as [112] put more emphasis on the agent platforms rather than the general methodology. In the aforementioned study, the platforms are discussed along with the variety of domains within which they have been applied, including examples from physics, chemistry, biology, cybersecurity, social modeling, economics, and the environment. However, as the emphasis is put on agent-based modeling and simulations, this work does not provide an in-depth analysis of platforms that could be applied in other fields. An example of a more recent study that follows this approach is [108,113].

One of the most recent articles ([114]) not only includes descriptions of the agent platforms but also summarizes the extensions created for them. Nevertheless, this contribution focuses primarily on the recently updated general-purpose platforms, thus providing only a partial overview of the current state of the art in the field. Furthermore, it also lacks information about platform licensing.

Another review of agent platforms is included as part of "The Handbook on Socially Interactive Agents" [115]. This work describes the research in the fields of social interactive agents, interactive virtual agents, and social robotics. As such, it contains a summary of agent platforms, with a particular focus on cognitive platforms, academic platforms, and commercial game engines. Since the book focuses primarily on the capabilities of SIAs, the landscape of platforms dedicated to other fields is not discussed. Additionally, the majority of the platforms included were developed many years ago; hence, some are no longer maintained. The latter is particularly true from the "2023 perspective".

While agent platforms have been evaluated in multiple publications, it can be noted that (1) most of the publications were focused on specific implementation domains and (2) the more general reviews were conducted several years ago, which, in view of the dynamics of the field, makes them no longer valid (nor, to some extent, relevant). One of the reasons for the lack of up-to-date, comprehensive reviews may be the maturity of the field itself. Due to the wide range of existing agent platforms, conducting an exhaustive but simultaneously coherent review requires a thorough selection of the platforms and prior establishment of assessment criteria. However, many existing platforms are not "easily accessible", which prolongs the time required for the (re)search and verification. Specifically, the most popular platforms are either the ones that were developed years ago (e.g., JADE) or the commercial platforms, which comprise only a small part of the agent platform domain. As such, this further complicates finding and evaluating newer or lesser-known platforms, which are often supported by individual developers or small research groups. Furthermore, in many cases, such platforms are introduced merely as proofs-of-concept, whereas the actual code is not fully implemented nor publicly available. Thus, they require additional verification to determine if they are accessible to users (and usable), which adds further complexity to the selection process. Furthermore, the assessment of the platforms may raise another issue. As the platforms are intended for different domains, they may employ very different architectural and technical approaches and, as such, be difficult to compare to each other. Hence, to provide readers with useful guidelines, this work started from a comprehensive search of the Web to (a) update existing information and (b) add platforms that have been recently introduced. Next, we found platforms that have been verified as to their actual usability. The results of this work are summarized in what follows.

4. General-Purpose Platforms

Let us start with the general-purpose agent platforms. These platforms are not intended for any particular type/area of application. They can be used for the implementation of various types of agent-based systems, including those dedicated to simulation and modeling [116]. Some of the platforms (e.g., JACK) simply incorporate into well-known programming languages (such as Java or Python) the concepts that are specifically related to agent systems (i.e., agents, messages, behaviors). In other cases, the presented agent platforms extend or are built upon already existing ones (e.g., JADEX, fjåge). The platforms included in what follows are both open-source and commercial. In order to make it more intuitive for the readers, they are described in separate sections.

4.1. Open-Source General-Purpose Platforms

The platforms presented in this section are open-source and can be used for the implementation of different kinds of agent systems. From the technical perspective, these platforms differ based on the programming language or adopted architectural design (i.e., some of them are distributive, whereas others are standalone). Furthermore, they incorporate various agent system-specific concepts, such as the BDI paradigm from the FIPA specification standards. Here, Table 1 first presents a summary of the selected open-source platforms. This is followed by additional information related to each of them.

- **ActressMas** —An agent framework inspired by the actor model and implemented using .NET asynchronous operations. One of its primary design objectives is conceptual simplicity and support for high-level agent abstractions, making it valuable for teaching purposes. In addition to actor-specific reactive behavior, it incorporates mechanisms for proactive behavior and, as such, can be used for the implementation of multi-agent systems. It supports the concept of agent mobility by means of agents residing in distributable containers. Additionally, it adheres to some of the FIPA specification standards for agent platforms, but it is not fully FIPA-compliant;
- **Akka**—A set of libraries dedicated to highly concurrent and distributed systems. In accordance with its design principles, the toolkit is scalable, fault-tolerant, resilient, and highly performative. Similarly to ActressMas, Akka is based on the actor model, where actors encapsulate states and behaviors. Here, the communication between entities is achieved in a message-driven fashion. The model is based on a hierarchical structure and the concept of actors' supervision (i.e., "parent" actors are the supervisors of task distribution and failure handlers). Furthermore, Akka systems are distributable on multiple nodes. It is worth pointing out that, since the beginning of 2023, Akka has been distributed under the BSL license. Therefore, it is free only for non-production usage. For production applications, it requires a commercial license;
- **Akka.NET**—Port of the original Akka to C# and F# for systems developed in .NET and Mono. Similarly to Akka, the toolkit is designed for highly concurrent and event-driven applications. The platform is based on high-level actor abstractions and asynchronous and distributed design. The architectural principles of the actor model are the same as in the original Akka platform. An extension provided for Akka.NET that is worth mentioning is the Phobos 2.0 monitoring tool. In particular, the tool enables tracking the system cluster and capturing different actor metrics (e.g., the actor's mailbox latency). In contrast to Akka, Akka.NET is fully open source and is released under the Apache 2.0 license;

- **ASTRA**—A Java-based agent-oriented programming (AOP) language for concurrent and distributed systems. The design of the language is based on simplicity and aims to resemble commonly known OOP languages (e.g., Java, C#). The platform is thus integrated with Java programming principles and, therefore, among other characteristics, it is strongly typed. On the side of AOP, ASTRA is inspired by AgentSpeak(L). As such, it incorporates all of its agent-related functionalities and, specifically, the concepts of the BDI paradigm (e.g., beliefs, events, and plan rules). Moreover, it also extends some parts of the traditional AgentSpeak(L) syntax; for instance, the syntax of plans. In terms of agent interaction, ASTRA provides two methods: (1) the message exchange (with the support of the FIPA ACL concept) and (2) resource sharing;
- **BDI4Jade**—A platform extending the agent-related functionalities of JADE (e.g., distributivity, agent communication) with BDI paradigm concepts. It provides means for the development of reasoning agents by including definitions of modularized reasoning strategies, goals, beliefs, and plans. In addition to that, the platform adopts the concept of capability relationships [117]. According to the information obtained from the platform's author, BDI4Jade is in a stable version and, as of now (at the time of writing), is being maintained without further plans to extend it with new features;
- **JaCaMo**—A platform for multi-agent programming consisting of three technologies: Jason, CArtAgO, and Moise. It is based on the JaCa programming model, which separates the programming for the logic of the agents from the programming for the environment. This separation of concerns is realized through a combination of different agent-based technologies, where (1) Jason facilitates the programming for the cognitive agents using the BDI approach, (2) CArtAgO enables the modeling of multi-agent virtual environments with artifacts, and (3) Moise allows for the definition of specifications for the agents' organizations. Moreover, JaCaMo can be supplemented with two extensions, JaCaMo Web and JaCaMo REST, that facilitate communication with the platform. JaCaMo REST provides an API for interaction with system components (i.e., agents and artifacts), whereas JaCaMo Web allows modifying agent instances at runtime;
- Java Agent Development Framework (**JADE**)—A Java-based industrial-grade framework dedicated to the implementation of multi-agent systems. Its primary objective is standard compliance and ,therefore, it is fully aligned with the FIPA specification. The communication between agents is undertaken in a peer-to-peer fashion through asynchronous message passing. JADE supports distributing agents on multiple containers running on different hosts. The agents are fully mobile, meaning that they can be moved from one container to another at system runtime. The platform offers a dedicated GUI and several debugging tools for monitoring the system. Besides the standalone functionalities, JADE is supported by several extensions, such as JADE Test Suite and Leap. Furthermore, it can be supplemented with an additional platform called WADE. It extends the original framework by aligning it with the workflow metaphor, where workflows are formalized as Java classes, which facilitates their use in the definition of system-internal logic;
- **JADEX**—A platform extending JADE with the principles of rational agents. The design of the platform follows the concepts of active components (ACs) and service component architecture (SCA). The JADEX components are able to both (1) act as passive service providers and (2) autonomously execute behaviors. Based on these features, they can be seen as the representation of the agents. The platform facilitates the cognitive BDI approach and the business process modeling notation workflow. It also supports mechanisms of dynamic agent discovery and mobility and allows the distribution of agents across multiple machines. The communication in JADEX is undertaken via the services and offers interaction protocols, including asynchronous message exchange;
- **Janus**—An agent platform implemented in SARL used in the creation of Web and desktop-based multi-agent applications. It can be used in both organizational-based

and holarchy-based agent systems. It incorporates the concepts of BDI architecture, allowing for the development of reasoning agents. In terms of agent interactions, it supports both synchronous and asynchronous communication protocols. The SARL language used in the implementation of Janus adheres to ACL accordingly and, in large part, to the FIPA specification. Furthermore, the platform also offers a set of monitoring tools that allow, for instance, monitoring of agents' activity (e.g., their beliefs and goals). Currently, the platform is included in the development of SARL. It has an active community and provides communication channels where developers can ask questions regarding encountered issues;

- **JS-son**—A versatile and extendable JavaScript library for the development of reasoning agents. The library supports the BDI architecture and adheres to reasoning loops within agents. From the conceptual perspective, JS-son agents can be programmed according to three different approaches: (1) a traditional BDI approach, (2) a simplified belief–plan approach, and (3) a goal-based approach. The first approach follows a traditional BDI architecture with concepts of the agent's beliefs, desires, and intentions. The belief–plan approach is a simplification of BDI where the agents use their perception of the environment and their internal state to execute plans through which they interact and update their own beliefs. Finally, the goal-based approach focuses strictly on the agent's goals, which are treated as the driving forces behind their behaviors. On the technical side, the platform enables programming distributable Web applications that can also be developed in a serverless manner (using Google Cloud functions);
- Smart Python Multi-Agent Development Environment (**SPADE**)—A multi-agent platform based on the XMPP/Jabber technology that offers features that assist in the construction of MASs. It is the first agent-based system that uses XMPP as its foundation. The communication between agents is achieved via the concept of communication channels, where the messages are dispatched using dedicated message templates. The platform facilitates the creation of agents (represented as users characterized with unique Jabber identifiers) and agent platforms (XMPP-based servers). It provides an extensible XML-based communication protocol that supports FIPA-ACL metadata. Furthermore, the platform offers a plugin (SPADE-BDI) that facilitates the incorporation of the BDI architecture into SPADE agents;
- Tuple Centres Spread over the Network (**TuCSoN**)—Java-based platform used for the development and coordination of multi-agent systems. Specifically, it is dedicated to model agents written in Java and tuProlog. Its main concept is based on tuple space coordination, where tuples (programmable in ReSpecT language) are data structures shared between and accessed by agents in order to facilitate their communication and coordination. The agents are able to interact in the tuple space through the use of primitives, which are inspired by LINDA's coordination model. In this context, TuCSoN extends the primitives of LINDA with its own types (e.g., "bulk" and "uniform"). Aside from the base project, several extensions of the platform are available, allowing integration with agents created using other agent platforms, such as JADE or JASON;
- **XKlaim—**a renewed and enhanced version of KLAIM, a coordination language for modeling and programming distributed systems. In particular, the extension considers the incorporation of high-level programming (Java-based) constructs. Similarly to TuCSoN (and the original KLAIM), the platform is based on the LINDA communication concepts and defines the simulations in tuple space. The nodes (agents) are organized in hierarchical nets. Furthermore, as XKlaim was designed to be used alongside the Java-based Klava library, it provides first-class abstractions known as localities (i.e., network addresses used to distribute the data among the agents), which can be distinguished into physical or logical types.

Table 1. Open-source general-purpose platforms.

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
1	ActressMas	3.0.0	C#	http://florinleon.byethost24.com/actressmas.html https://github.com/florinleon/ActressMas https://web.archive.org/web/20230108225119/https://github.com/florinleon/ActressMas	https://github.com/florinleon/ActressMas/blob/master/LICENSEhttps://web.archive.org/web/20230610015659/https://github.com/florinleon/ActressMas/blob/master/LICENSE	Platform that can be used for teaching multi-agent systems, includes implementations of popular multi-agent protocols and algorithms
2	Akka	2.8.2	Scala/Java	https://akka.io/ http://web.archive.org/web/20230606075845/https://akka.io/ https://github.com/akka/akka https://web.archive.org/web/20230321070311/https://github.com/akka/akka	https://github.com/akka/akka/blob/main/LICENSE https://web.archive.org/web/20230121153534/https://github.com/akka/akka/blob/main/LICENSE https://www.lightbend.com/akka/license-faq	Message-driven actor model-based platform with a hierarchical structure for highly concurrent, distributed applications
3	Akka.NET	1.5.7	C#/F#	https://getakka.net/ https://web.archive.org/web/20230525020504/https://getakka.net/ https://github.com/akkadotnet/akka.net https://web.archive.org/web/20230511055116/https://github.com/akkadotnet/akka.net	https://github.com/akkadotnet/akka.net/blob/dev/LICENSE https://web.archive.org/web/20230511110703/https://github.com/akkadotnet/akka.net/blob/dev/LICENSE	Port of the Akka platform to .NET that extends Akka with C# and F# capabilities
4	ASTRA	1.3.4	Java	http://guide.astralanguage.com/en/latest/ https://web.archive.org/web/20230208192659/http://guide.astralanguage.com/en/latest/ https://gitlab.com/astra-language https://web.archive.org/web/20230610020529/https://gitlab.com/astra-language	https://gitlab.com/astra-language/astra-core/-/blob/master/LICENSE https://web.archive.org/web/20230610020640/https://gitlab.com/astra-language/astra-core/-/blob/master/LICENSE	Agent-oriented programming language for distributed and concurrent systems

Table 1. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
5	BDI4Jade	2.0	Java	https://github.com/ingridnunes/bdi4jade https://web.archive.org/web/20230610021150/https://github.com/ingridnunes/bdi4jade	https://github.com/ingridnunes/bdi4jade/blob/main/LICENSE https://web.archive.org/web/20230610021517/https://github.com/ingridnunes/bdi4jade/blob/main/LICENSE	BDI architecture with reasoning cycle implemented on top of the JADE platform
6	JaCaMo	1.1	AgentSpeak (Jason)	http://jacamo.sourceforge.net/ https://web.archive.org/web/20230610021817/https://jacamo.sourceforge.net/ https://github.com/jacamo-lang/jacamo https://web.archive.org/web/20230610022031/https://github.com/jacamo-lang/jacamo https://sourceforge.net/projects/jacamo/ https://web.archive.org/web/20230610022110/https://sourceforge.net/projects/jacamo/	https://github.com/jacamo-lang/jacamo/blob/master/LICENSE https://web.archive.org/web/20230610022154/https://github.com/jacamo-lang/jacamo/blob/master/LICENSE	BDI-based platform combining Jason, CArtAgO, and Moise used in programming multi-agent systems
7	JADE	4.6.0	Java	https://jade.tilab.com/ https://web.archive.org/web/20230610022307/https://jade.tilab.com/	https://www.gnu.org/licenses/old-licenses/lgpl-2.0.en.html https://web.archive.org/web/20230610022420/https://www.gnu.org/licenses/old-licenses/lgpl-2.0.en.html	FIPA-compliant agent-based framework with graphical debugging tools used in implementing distributed MASs
8	JADEX	4.0.267	Java	https://www.activecomponents.org/#/project/news https://web.archive.org/web/20230610022551/https://www.activecomponents.org/ https://github.com/actoron/jadex https://web.archive.org/web/20230610022702/https://github.com/actoron/jadex	https://github.com/actoron/jadex/blob/master/LICENSE https://web.archive.org/web/20230610022725/https://github.com/actoron/jadex/blob/master/LICENSE	Service component architecture (SCA) platform with BDI architecture that extends JADE with rational agents

Table 1. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
9	Janus	3.0.12.0	Java/SARL	http://www.sarl.io/runtime/janus/ https://web.archive.org/web/20230610022810/http://www.sarl.io/runtime/janus/ https://github.com/sarl/sarl https://web.archive.org/web/20230610022813/https://github.com/sarl/sarl	https://github.com/sarl/sarl/blob/master/LICENSE https://web.archive.org/web/20230610022947/https://github.com/sarl/sarl/blob/master/LICENSE	Platform implemented in SARL used to develop, run, monitor, and display agent-based applications
10	JS-son	0.0.15	JavaScript	https://github.com/TimKam/JS-son https://web.archive.org/web/20230610022951/https://github.com/TimKam/JS-son https://js-son.readthedocs.io/en/latest/ https://web.archive.org/web/20230610023042/https://js-son.readthedocs.io/en/latest/	https://github.com/TimKam/JS-son/blob/master/LICENSE https://web.archive.org/web/20230610023315/https://github.com/TimKam/JS-son/blob/master/LICENSE	JavaScript agent platform with a lightweight and extensible design employing the BDI approach and reasoning agent loops
11	SPADE	3.2.3	Python	https://pypi.org/project/spade/ https://web.archive.org/web/20230610023514/https://pypi.org/project/spade/ https://github.com/javipalanca/spade https://web.archive.org/web/20230610023525/https://github.com/javipalanca/spade https://spade-mas.readthedocs.io/en/latest/ https://web.archive.org/web/20230610023619/https://spade-mas.readthedocs.io/en/latest/	https://github.com/javipalanca/spade/blob/master/LICENSE https://web.archive.org/web/20230610023635/https://github.com/javipalanca/spade/blob/master/LICENSE	A multi-agent platform based on the instant messaging XMPP/Jabber technology with support from FIPA metadata

Table 1. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
12	TuCSoN	0.2.9	Java	https://apice.unibo.it/xwiki/bin/view/TuCSoN/ https://web.archive.org/web/20230610023750/https://apice.unibo.it/xwiki/bin/view/TuCSoN/ https://github.com/TuCSoN-Coord/TuCSoN/ https://web.archive.org/web/20230610023754/https://github.com/TuCSoN-Coord/TuCSoN/	https://github.com/TuCSoN-Coord/TuCSoN/blob/master/LICENSE https://web.archive.org/web/20230610023918/https://github.com/TuCSoN-Coord/TuCSoN/blob/master/LICENSE	Java library to coordinate agents using programmable tuple centers
13	XKlaim	2.4.0	Java	https://link.springer.com/chapter/10.1007/978-3-030-21485-2_8 https://web.archive.org/web/20230611173416/https://link.springer.com/chapter/10.1007/978-3-030-21485-2_8 https://github.com/LorenzoBettini/xklaim https://web.archive.org/web/20230610024056/https://github.com/LorenzoBettini/xklaim	https://github.com/LorenzoBettini/xklaim/blob/master/LICENSE https://web.archive.org/web/20230610024136/https://github.com/LorenzoBettini/xklaim/blob/master/LICENSE	A coordination language for modeling and programming distributed systems addressing usability concerns

In terms of the presented platforms, there are no restrictions on the potential domains in which they may be applicable. However, it needs to be emphasized that, when considering the optimization aspects of a system, they may not be the most suitable choices. This is particularly relevant for systems that are intended to facilitate large-scale simulations, such as biological simulations or models of population phenomena. Despite the fact that such systems could possibly be implemented using the platforms mentioned above, it would require considerable effort from the developer to make the final application reliable. Thus, in such cases, it would be better to consider platforms that have been designed and implemented with the scalability of the system in mind. In this context, let us describe open-source platforms that can be seen as devoted to agent-based modeling and simulation.

4.2. Open-Source General Simulations and Modeling Platforms

The presented platforms range from those used in small-scale simulations to those that can be used in large-scale simulations. Several of them rely on parallel computing, specifically utilizing graphics processing units (GPUs) or clusters, to optimize their overall efficiency. Some of the platforms are extensions of their predecessors; for instance, FLAME GPU 2 is an extension of FLAME. A common factor that could be noted for these platforms is that most of them come with additional tools that assist users with the visualization of prepared models. The individual features of the presented platforms are summarized in Table 2. As previously, this is followed by a more detailed discussion of each platform.

Table 2. Open-source general-purpose modeling platforms.

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
1	Agents Assembly	0.0.56	AASM	https://agents-assembly.com/ https://web.archive.org/web/20230610025944/https://agents-assembly.com/ https://github.com/agent-based-information-flow-simulation https://web.archive.org/web/20230610025942/https://github.com/agent-based-information-flow-simulation https://link.springer.com/chapter/10.1007/978-3-031-18192-4_42 https://web.archive.org/web/20230611173556/https://link.springer.com/chapter/10.1007/978-3-031-18192-4_42	https://github.com/agent-based-information-flow-simulation/agents-assembly-translator/blob/main/LICENSE.md https://web.archive.org/web/20230610030155/https://github.com/agent-based-information-flow-simulation/agents-assembly-translator/blob/main/LICENSE.md	Domain-specific language that is intended for scalable containerized simulations
2	AgentPy	0.1.5	Python	https://agentpy.readthedocs.io/en/latest/ https://web.archive.org/web/20230610030332/https://agentpy.readthedocs.io/en/latest/ https://github.com/JoelForamitti/agentpy https://web.archive.org/web/20230610030502/https://github.com/JoelForamitti/agentpy	https://github.com/JoelForamitti/agentpy/blob/master/LICENSE https://web.archive.org/web/20230610030459/https://github.com/JoelForamitti/agentpy/blob/master/LICENSE	Platform for ABM, useful in data analysis in single environments
3	Agents.jl	5.13.0	Julia	https://juliadynamics.github.io/Agents.jl/stable/ https://web.archive.org/web/20230610034206/https://juliadynamics.github.io/Agents.jl/stable/ https://github.com/JuliaDynamics/Agents.jl https://web.archive.org/web/20230610034231/https://github.com/JuliaDynamics/Agents.jl	https://github.com/JuliaDynamics/Agents.jl/blob/main/LICENSE.md https://web.archive.org/web/20230610034321/https://github.com/JuliaDynamics/Agents.jl/blob/main/LICENSE.md	Platform that relies on grid-based environments for 1D, 2D, and 3D distributed simulations
4	AgentScript	0.10.19	JavaScript	https://agentscript.org/ https://web.archive.org/web/20230610034324/https://agentscript.org/ https://github.com/backspaces/agentscript https://web.archive.org/web/20230610034448/https://github.com/backspaces/agentscript	https://github.com/backspaces/agentscript0/blob/master/LICENSE https://web.archive.org/web/20230610034450/https://github.com/backspaces/agentscript0/blob/master/LICENSE	Platform based on NetLogo semantics with MVC architecture
5	DEVS-Suite Simulator	7.0	Java	https://acims.asu.edu/devs-suite/ https://web.archive.org/web/20230610034609/https://acims.asu.edu/devs-suite/ https://sourceforge.net/projects/devs-suitesim/ https://web.archive.org/web/20230610034623/https://sourceforge.net/projects/devs-suitesim/	https://www.gnu.org/licenses/lgpl-3.0.en.html https://web.archive.org/web/20230610034654/https://www.gnu.org/licenses/lgpl-3.0.en.html	Platform based on Parallel DEVS simulator that uses modeling with cellular automata

Table 2. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
6	EcoLab	5.77	C++	https://ecolab.sourceforge.net/ https://web.archive.org/web/20230610034728/https://ecolab.sourceforge.net/ https://github.com/highperformancecoder/ecolab https://web.archive.org/web/20230610034827/https://github.com/highperformancecoder/ecolab	https://github.com/highperformancecoder/ecolab/blob/master/LICENSE https://web.archive.org/web/20230610034918/https://github.com/highperformancecoder/ecolab/blob/master/LICENSE	Agent-based simulation platform with advanced data structure and algorithms
7	fjåge	1.10.5	Java, Groovy	https://fjage.readthedocs.io/en/latest/index.html https://web.archive.org/web/20230610034949/https://fjage.readthedocs.io/en/latest/index.html https://github.com/org-arl/fjage https://web.archive.org/web/20230610035055/https://github.com/org-arl/fjage	https://github.com/org-arl/fjage/blob/master/LICENSE.txt https://web.archive.org/web/20230610035056/https://github.com/org-arl/fjage/blob/master/LICENSE.txt	JADE-based platform for real-time and discrete event simulations
8	FLAME GPU 2	2.0.0	CUDA/C++ Python	https://flamegpu.com/ https://web.archive.org/web/20230610035051/https://flamegpu.com/ https://github.com/FLAMEGPU/FLAMEGPU2 https://web.archive.org/web/20230610035200/https://github.com/FLAMEGPU/FLAMEGPU2	https://github.com/FLAMEGPU/FLAMEGPU2/blob/master/LICENSE.md https://web.archive.org/web/20230610035211/https://github.com/FLAMEGPU/FLAMEGPU2/blob/master/LICENSE.md	GPU-based extended version of FLAME supporting CUDA (C++) and Python interfaces
9	GAMA	1.8.1	Java/GAML	https://gama-platform.org/ https://web.archive.org/web/20230610035210/https://gama-platform.org/ https://github.com/gama-platform/gama https://web.archive.org/web/20230610035421/https://github.com/gama-platform/gama	https://github.com/gama-platform/gama/blob/GAMA_1.9.2/LICENSE https://web.archive.org/web/20230610035718/https://github.com/gama-platform/gama/blob/GAMA_1.9.2/LICENSE	Multi-agent platform for spatially explicit simulations
10	InsightMaker	2.0.0	Modeling with Web UI	https://insightmaker.com/ https://web.archive.org/web/20230610035432/https://insightmaker.com/ https://github.com/scottfr/simulation https://web.archive.org/web/20230610035623/https://github.com/scottfr/simulation	https://github.com/scottfr/simulation/blob/main/LICENSE https://web.archive.org/web/20230610035721/https://github.com/scottfr/simulation/blob/main/LICENSE	Simulation environment that supports ABM in the browser
11	JAS-Mine	4.1.0	Java	https://www.microsimulation.ac.uk/jas-mine/ https://web.archive.org/web/20230610035827/https://www.microsimulation.ac.uk/jas-mine/ https://github.com/jasmineRepo https://web.archive.org/web/20230610035838/https://github.com/jasmineRepo	https://www.gnu.org/licenses/old-licenses/lgpl-2.0.en.html https://web.archive.org/web/20230610022420/https://www.gnu.org/licenses/old-licenses/lgpl-2.0.en.html	Platform for discrete-event simulations with agent-based and microsimulation models

Table 2. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
12	JSimpleSim	3.0.0	Java	https://jsimplesim.org/ https://web.archive.org/web/20230610035955/https://jsimplesim.org/ https://github.com/simnation/JSimpleSim https://web.archive.org/web/20230610040029/https://github.com/simnation/JSimpleSim	https://github.com/simnation/JSimpleSim/blob/master/LICENSE.md https://web.archive.org/web/20230610040039/https://github.com/simnation/JSimpleSim/blob/master/LICENSE.md	Java-based simulation and modeling framework with the discrete-event approach (DES)
13	MASON	21	Java	https://cs.gmu.edu/~eclab/projects/mason/ https://web.archive.org/web/20230610040105/https://cs.gmu.edu/~eclab/projects/mason/ https://github.com/eclab/mason/ https://web.archive.org/web/20230610040122/https://github.com/eclab/mason/	https://github.com/eclab/mason/blob/master/LICENSE https://web.archive.org/web/20230610040146/https://github.com/eclab/mason/blob/master/LICENSE	Discrete-event ABM simulation core with 2D or 3D visualization
14	MASS	1.3.1	C++, CUDA, Java	http://depts.washington.edu/dslab/MASS/ https://web.archive.org/web/20230610040155/http://depts.washington.edu/dslab/MASS/ https://bitbucket.org/mass_library_developers/	Open source (Java and C++)	Platform for parallel multi-agent and spatial simulation
15	Mesa	1.2.0	Python3	https://github.com/projectmesa/mesa https://web.archive.org/web/20230610040256/https://github.com/projectmesa/mesa https://www.researchgate.net/publication/328774079_Mesa_An_Agent-Based_Modeling_Framework https://web.archive.org/web/20221216123114/https://www.researchgate.net/publication/328774079_Mesa_An_Agent-Based_Modeling_Framework	https://github.com/projectmesa/mesa/blob/main/LICENSE https://web.archive.org/web/20230610040411/https://github.com/projectmesa/mesa/blob/main/LICENSE	Python ABM platform, an alternative to NetLogo or Repast
16	MOOSE	2022-06-10	C++	https://mooseframework.inl.gov/ https://web.archive.org/web/20230610040527/https://mooseframework.inl.gov/ https://github.com/idaholab/moose https://web.archive.org/web/20230610040531/https://github.com/idaholab/moose	https://github.com/idaholab/moose/blob/next/LICENSE https://web.archive.org/web/20230610040539/https://github.com/idaholab/moose/blob/next/LICENSE	Parallel multi-physics object-oriented simulation platform
17	NetLogo	6.3.0	Scala, Java	https://ccl.northwestern.edu/netlogo/index.shtml https://web.archive.org/web/20230610040628/https://ccl.northwestern.edu/netlogo/index.shtml https://github.com/NetLogo/NetLogo https://web.archive.org/web/20230610040639/https://github.com/NetLogo/NetLogo	https://www.gnu.org/licenses/old-licenses/gpl-2.0.html https://web.archive.org/web/20230610023727/https://www.gnu.org/licenses/old-licenses/gpl-2.0.html	Modeling environment for ABM natural and social simulations

Table 2. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
18	Repast Suite	2.3.1 2.10.0 1.1.1	C++, Java Python	https://repast.github.io/ https://web.archive.org/web/20230610040737/https://repast.github.io/ https://github.com/Repast https://web.archive.org/web/20230610040755/https://github.com/Repast	https://repast.github.io/license.html https://web.archive.org/web/20230610040753/https://repast.github.io/license.html	Distributive ABM platform for computing clusters, workstations, and supercomputers
19	SIMILAR	1.0.0	Java	https://www.lgi2a.univ-artois.fr/~morvan/similar.html https://web.archive.org/web/20230610040839/https://www.lgi2a.univ-artois.fr/~morvan/similar.html https://github.com/gildasmorvan/similar https://web.archive.org/web/20230610040900/https://github.com/gildasmorvan/similar	https://github.com/gildasmorvan/similar/blob/master/LICENSE.txt https://web.archive.org/web/20230610040926/https://github.com/gildasmorvan/similar/blob/master/LICENSE.txt	Multi-level ABM meta-model with influence reaction model
20	SpaDES	2.0.9	R	https://spades.predictiveecology.org/ https://web.archive.org/web/20230610041222/https://spades.predictiveecology.org/ https://github.com/PredictiveEcology/SpaDES https://web.archive.org/web/20230610030332/https://agentpy.readthedocs.io/en/latest/	https://github.com/PredictiveEcology/SpaDES/blob/master/LICENSE http://web.archive.org/web/20230610041304/https://github.com/PredictiveEcology/SpaDES/blob/master/LICENSE	Package for event-based models with spatially explicit models

- **Agents Assembly**—A domain-specific language based on assembly mnemonics for implementing multi-agent systems. It is part of a toolset intended for large-scale simulations in containerized environments that aims to streamline the development of agent systems for users without specialized programming skills. The language supports programming constructs that are directly related to agent systems (i.e., agents, behaviors, messages) and offers various mathematical statements and concepts commonly used in other programming languages. For the visualization of created systems, the AASM provides a dedicated graph modifier that also supports the creation of graphs based on statistical distributions. The language is translated in two steps: first into a target-agnostic intermediate representation and then into Python, based on the SPADE platform. AASM offers a dedicated GUI for simulation definition and management, a run environment, and an XMPP-based communication server. As the platform enables running SPADE instances in a containerized (dockerized) manner, it is highly scalable and, as such, well-suited for large-scale simulations;
- **AgentPy**—A platform that facilitates agent-based modeling. It is designed strictly for scientific use; therefore, it provides tools for parameter sampling, Monte Carlo experiments, stochastic processes, and sensitivity analysis, among others. The implemented models are built using agents that are combined in groups called sequences. Such an approach allows the manipulation of multiple entities at once. Furthermore, they are placed in environments according to specified positions. Currently, three types of environment topologies are supported: (1) grids, (2) spaces, and (3) networks. The platform enables the graphical representation of simulations using grid plots and interactive visualization within Jupyter Notebooks. Additionally, AgentPy also offers an experiment class that can facilitate the parallel processing of simulations;

- **Agents.jl**—A framework for Julia that supports the creation of agent-based models (ABMs). It is based on a modular and function-based design and, as such, is easily extendable. The platform supports the implementation of different types of agents, which can also be generated using Agents.jl macros. The simulations are conducted in structured spaces of four types: grids, graphs, continuous spaces, and OpenStreetMaps. The framework provides tools for collecting the simulations' aggregated data and presenting them visually. Furthermore, it allows monitoring the process of modeling using an interactive "Data Voyager" tool. The simulations can be executed concurrently on numerous cores. In comparison to other ABM platforms, Agents.jl offers higher memory capacity, and it is restricted only by hardware limits. Additionally, it provides N-dimensional space;
- **AgentScript**—An agent-based modeling framework that incorporates the semantics of NetLogo and relies on the model/view/control (MVC) architecture. As such, its design is based on the separation of concerns. Specifically, the system is divided into three parts: the first (model) defines the semantics of agents (turtles), connections (links), and patches; the second (view) represents the model graphically using a 2D/3D canvas (with support for GISs or geo-modules); and the third (control) provides functionalities allowing interaction with the system via a dedicated graphical user interface. By relying on the Javascript/Coffeescript programming language, the platform aims to be highly deployable. Furthermore, it provides an online interface for developers to explore and learn how to work with the platform;
- **DEVS-Suite**—A simulator that is based on the parallel DEVS component and the cellular automata approach. It automates the experimentation process and creates time data trajectories in real time. Furthermore, it features hierarchical model libraries and can be used to animate implemented models. As such, the platform provides the interfaces for both separate and stacked time trajectories, which can be used in component tracking. In terms of cellular automata, DEVS-Suite offers independent tracking of any number of cells with distinct start times. The simulator contains "Models" divided into "CellularAutomata" (e.g., game of life, system biology chemotaxis, forest fire) and "Component" (e.g., single-input single-output, basic, and multiprocessor architectures) packages. However, these packages are not limited, and users can also add their own model packages alongside the provided ones;
- **EcoLab**—An agent-based simulation framework that facilitates experiment-oriented metaphors. Initially, it was designed for an abstract ecology model, but as the environment evolved and more types of models were developed using it, it expanded to become a general-purpose platform. As EcoLab is implemented in C++, it incorporates several advanced algorithms and data structures. The models implemented with the platform can be modified and controlled dynamically at runtime. Additionally, EcoLab supports creating models of distributed agent topologies that can run on the MPI-based cluster. It also facilitates the checkpoint mechanisms, which make it possible to store the model's state and periodically report it to the visualization client. As such, the models can be accessed via a dedicated GUI with graph and histogramming features;
- **fjåge**—A lightweight platform for the development of multi-agent systems. It aims to be fast and easy to understand. The implemented agents are associated with unique identifiers, which facilitate the exchange of messages. Furthermore, it implements various types of agent actions and behaviors. The primary features of the platform were inspired by the JADE framework. Similarly to JADE, fjåge supports the distribution of agents across multiple containers. However, unlike JADE, the concept of container-running platforms is extended by introducing two types of container management: real-time and event-driven. In this way, fjåge facilitates simulation development and is suitable for rapid testing. The core difference between JADE and fjåge is that, while fjåge supports the concepts of FIPA, it cannot communicate with other FIPA-compliant agent-based systems;

- **FLAME GPU 2—**The second version of the GPU extension to the FLAME framework. It incorporates agent-related concepts into C and CUDA code. The agents implemented in FLAME GPU 2 are associated with specific types (corresponding to given models) and encapsulate the state in the form of sets of behaviors. The communication is undertaken in a message-driven manner, and each message, similarly to agents, is also identified by the corresponding type, which defines a given communication strategy. Furthermore, the method and scope of interaction between agents and the environment are defined through agent functions. The simulations run in FLAME GPU 2 can be executed individually or in a batched manner. Due to the fact that the data are situated on GPUs, the platform offers the ability to visualize a large number of agents. Additionally, the second version of the platform provides a Python-based interface for writing models;
- **GAMA—**A simulation platform that facilitates the development of spatially explicit multi-agent simulations, including large-scale simulations. The model is based on a general approach and can therefore be used in a variety of applications. Additionally, in order to meet more specific needs, the platform incorporates several plugins (e.g., Remote.GUI for use in participative simulations). The GAMA platform uses a dedicated language called GAML, the design objective of which is to make it intuitive and easy to use by non-computer scientists. Agents in GAMA can be created from any dataset, including geographic information system (GIS) data. Furthermore, it also provides easy agent inspection, allows the creation of user-controlled action panels, and provides multi-layer 2D and 3D visualizations;
- **Insight Maker—**A modeling framework that employs multimethod models. The platform is capable of mapping conceptual models onto system descriptions through loop diagrams or pictures prepared in a dedicated diagramming tool. The prepared models can then be extended by adding dynamics using one of two modeling paradigms provided by the platform. The first one is called System Dynamics and focuses on the high-level behavior of the system, where the population is treated as a whole. The second approach involves agent-based modeling and allows the creation of separate agents in order to explore interactions between individuals. The diagramming also provides the developers with the ability to share their models publicly;
- **JAS-Mine—**A platform that facilitates discrete-event simulations and the development of large-scale data-driven models. It is designed to follow the separation principle, where the data management is decoupled from its representation and is based on a three-layer architecture consisting of the model (i.e., implementation of simulation components), collector (i.e., processing of simulation statistics), and observer (i.e., simulation visualization). The simulation is handled and organized by the "scheduler" component, which is common to all agents and responsible for ordering the simulation events. Among its features, JAS-Mine also provides integrated I/O communication services (e.g., RDBMS and automatic CSV table creation), and it includes an advanced multi-run utility that supports experiment design. Furthermore, the platform implements regression libraries that assist in the uncertainty analysis of the model outcome;
- **JSimpleSim—**A Java-based simulation and modeling framework that employs a discrete-event approach. It aims to facilitate the building of agent-based models while also supporting fast simulations. The models are interpreted as grids that embed the agents, with behaviors defined in an event-driven fashion (i.e., agents contain queues listing all of their events). The communication between agents employs two concepts: direct messaging and routing, where the messages are sent via dedicated ports. From the architectural perspective, the platform is characterized by the separation of simulation from modeling. Therefore, models prepared using the platform can be run on another simulator. Additionally, JSimpleSim supports sequential and concurrent simulation execution. It also offers, for the developers, diverse models, including cellular automata, hierarchical models, and mesh models;

- Multi-Agent Simulator of Neighborhoods/Networks (**MASON**)—A Java-based, discrete-event, multi-agent simulation platform for lightweight, large-scale simulations. It offers model libraries and visualization tools for both 2D and 3D applications. It is dedicated to multi-agent simulations with numerous agents that run on a single machine. Furthermore, it is complemented by several extensions, including GeoMason (support for GIS), Distributed MASON (cluster computing using MPI, cloud computing, and distributed applications), and D-MASON (the stable predecessor of Distributed Mason). Even though the latest platform release was at the end of 2022, the platform is still under continuous development. The information regarding the latest MASON versions is available only on the official platform's website and is no longer included in the code repository;
- **MASS**—A library designed for parallelizing multi-agent and spatial simulations, enabling the modeling of emergent collective behavior. It addresses the parallelization challenges in prevalent shared-memory approaches by offering a parallel-computing capability for multi-agent and spatial simulations across a cluster of computing nodes. The simulation entities are represented by the distributed arrays of agents. The communication between these agents is undertaken through periodic data exchange. The agents can communicate with both the agents residing in the same place as well as the agents from other places. Furthermore, the platform supports agent mobility by facilitating the migration of agents to remote nodes;
- **Mesa**—A modular framework used for the development of agent-based models. It is composed of modules that are organized into three categories: modeling, analysis, and visualization. The modeling modules define structures used in the development of agents. In particular, they include classes of agents, a definition of agents' movement space (which can be seen as a grid), and a scheduler used to organize actions (which can also be added at runtime). Analysis modules provide tools for data collection and the capability of running the model with various parameter settings. Finally, the visualization modules include classes for interactive model visualizations through a JavaScript-based server;
- Multiphysics Object-Oriented Simulation Environment (**MOOSE**)—A finite-element, multiphysics framework that provides a high-level interface for nonlinear solver technology. Its design aims for it to be aligned with real-life problems and provide a modular and extendable architecture. As such, it offers a straightforward API. Among its other features, the platform provides fully coupled and fully implicit multiphysics solvers. Additionally, it supports automatic parallelization and, in particular, it enables runs that utilize over 100,000 CPU cores. Such a dimension-independent, parallel-geometric search feature makes it suitable for contact-related applications. Furthermore, MOOSE offers built-in discontinuous Galerkin (DG) methods and parallel multiscale solutions;
- **NetLogo**—A modeling platform that is suitable for the representation of complex dynamic systems. In particular, it can facilitate the simulation of natural and social phenomena. Through the utilization of thousands of autonomous "agents", the platform allows exploring linkages between the micro-level behavior of individuals and the macro-level patterns of their collective interactions. In particular, NetLogo agents are grouped into "patches" (stationary agents), "turtles" (mobile agents), and "links" (connections between agents). The platform provides utilities that allow monitoring and modifying agents at runtime. It also offers a set of visualization tools, including plots (e.g., line plots, bar plots, etc.) and model visualization in 2D or 3D. Furthermore, NetLogo can be run using HubNet, which enables participatory simulations that can be used in teaching agent systems. Here, HubNet makes it possible to connect and control the system using individual student devices;
- **Repast Suite—**A collection of agent-based platforms that consists of three open-source frameworks: **Repast Symphony**, **Repast HPC**, and **Repast4Py**:

(1) **Repast Symphony** is a Java-based toolkit that offers features for agent-based modeling. It allows development using ReLogo (a dialect of Logo), Groovy, or Java, as well as including a pure Java point-and-click execution environment with built-in logging and graphing utilities. The platform offers a hierarchically nested definition of space. It allows for the visualization of 2D and 3D environments, networks (with the JUNG network modeling library), and geographical spaces (GIS support). Furthermore, Repast incorporates a concurrent, multi-threaded discrete event scheduler and provides a set of built-in statistical and numerical methods, including random number generation, statistical distributions, and specialized mathematics;
(2) **Repast High-Performance Computing (HPC)** is the C++ version of Repast Symphony optimized and rewritten from its previously created Java-based version. It incorporates the fundamental concepts of its predecessor, such as contexts and projections, while optimizing them to function in a parallel-distribution manner. It was specifically designed for utilization with large computing clusters and supercomputers;
(3) **Rephast4Py** is the latest addition to the Repast Suite and is built on the foundation of Repast HPC. It offers features and functionalities facilitated by the Python programming language. In contrast to its high-performance counterpart, Repast4Py prioritizes accessibility and ease of use. Its distributed agent-based modeling capability enables the development of complex system models that capture the scale and relevant details of many social problems;

- **SIMILAR—**Agent-based modeling (ABM) meta-model based on the influence reaction model (IRM4MLS). It is based on two-step action processing, where firstly agents generate "influences" (i.e., decisions made based on the agent's internal state) and then the system "reacts" (i.e., evaluating effects according to the environment's state). It is dedicated to complex systems that use knowledge from multiple viewpoints. The platform includes a generic and modular formal model and a simulation API preserving the formal model structure. To enhance efficiency, the API suite is separated between two kernels (a micro-kernel and extended kernel), which the developers may fine-tune according to their needs. Furthermore, another SIMILAR-based platform exists, Similar2Logo, which employs a logo-like approach. It serves as an example of how the SIMILAR platform can be employed in practice and is being used for educational purposes;
- **SpaDES—**An R meta-package that facilitates the implementation of event-based models, particularly spatial models (i.e., raster-based, event-based, and agent-based models). Its main objective is modularity, through which the users can extend the platform with additional functionalities. The SpaDES core is based on discrete event simulation (DES). The platform is composed of modules represented by structured R scripts that utilize event scheduling and offer distinct semi-autonomous algorithms. The dependencies of modules, individual parameters, and input and output data are defined in module metadata. From the architectural perspective, the platform follows the bottom-up approach, where the model is set up based on the modules' dependencies. Additionally, SpaDES offers several visualization tools and caching methods.

All the platforms considered so far are open-source, and their code can be easily accessed and modified depending on individual needs. The development of such platforms is also strongly supported by the developers' communities (e.g., GAMA, Mesa), which, in some cases, contribute substantially to their growth. Let us now move to a summary of the general-purpose platforms with closed code sources that are distributed under commercial licenses.

4.3. Commercial Platforms

The next section includes both general and simulation-oriented platforms that require buying a commercial license. Wherever possible, the links to the licensing agreements are provided in the table. It is worth stressing that some of the platforms offer special discounts for research or academia. Hence, readers interested in pursuing them for educational purposes are encouraged to examine this matter further. The platforms are listed in Table 3. Following that, additional descriptions are provided.

Table 3. General-purpose commercial platforms.

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
1	AnyLogic	8.8.2	Java	https://www.anylogic.com/ https://web.archive.org/web/20230610042149/https://www.anylogic.com/ https://www.youtube.com/channel/UCdH-e29FvfphfWmI2EMZPhg https://web.archive.org/web/20230610042556/https://www.youtube.com/channel/UCdH-e29FvfphfWmI2EMZPhg	Commercial (https://www.anylogic.com/upload/license_agreements/software-licensing-agreement-for-anylogic.pdf https://web.archive.org/web/20230610042209/https://www.anylogic.com/upload/license_agreements/software-licensing-agreement-for-anylogic.pdf), with free version for academia	General-purpose commercial simulation software with multimethod modeling
2	ExtendSim	10.0.9	C++	https://extendsim.com/ https://web.archive.org/web/20230610042456/https://extendsim.com/	Commercial (https://extendsim.com/images/downloads/forms/licenseAgreement/License_Agreement.pdf https://web.archive.org/web/20230610042310/https://extendsim.com/images/downloads/forms/licenseAgreement/License_Agreement.pdf)	Software tools that use an agent-based modeling methodology for business purposes
3	FlexSim	23.1.0	C++	https://www.flexsim.com/ https://web.archive.org/web/20230610042528/https://www.flexsim.com/	Commercial (https://assets.flexsim.com/eula/ https://web.archive.org/web/20230610042603/https://assets.flexsim.com/eula/)	Discrete event-based 3D simulation modeling and analysis software
4	FlexSim Hc	23.1.0	C++	https://www.flexsim.com/healthcare/flexsim-hc/ https://web.archive.org/web/20230610042729/https://www.flexsim.com/pl/healthcare/flexsim-hc/	Commercial (https://assets.flexsim.com/eula/ https://web.archive.org/web/20230610042603/https://assets.flexsim.com/eula/)	3D simulation environment for the analysis of healthcare facilities
5	GoldSim	14.0.R2	C++	https://www.goldsim.com/ https://web.archive.org/web/20230610042816/https://www.goldsim.com/ https://media.goldsim.com/Documents/Manuals/GoldSim.pdf https://web.archive.org/web/20230610042756/https://media.goldsim.com/Documents/Manuals/GoldSim.pdf	Commercial (https://media.goldsim.com/Documents/Software/GoldSim_EULA.pdf https://web.archive.org/web/20230610042816/https://media.goldsim.com/Documents/Software/GoldSim_EULA.pdf), free options for academia	Platform for dynamic simulation and visualization of complex systems in science and business
6	JACK	5.6 (jack56d)	Java	https://aosgrp.com.au/jack/ https://web.archive.org/web/20230610043000/https://aosgrp.com.au/jack/ https://aosgrp.com.au/research/ https://web.archive.org/web/20230610042937/https://aosgrp.com.au/research/	Commercial license (proprietary)	Environment for commercial-grade multi-agent systems that employs the BDI architecture

Table 3. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
7	Simio	15.240	GUI-based programming	https://www.simio.com/software/simulation-software.php https://web.archive.org/web/20230610043021/https://www.simio.com/software/simulation-software.php	Commercial license, some free options for academic use	Agent-based software supporting continuous process and discrete event systems, used to conduct real-time risk analysis
8	Simudyne	2.5	Java	https://www.simudyne.com/technology/agent-based-modeling/ https://web.archive.org/web/20230610043207/https://www.simudyne.com/technology/agent-based-modeling/ https://docs.simudyne.com/overview/welcome https://web.archive.org/web/20230610043220/https://docs.simudyne.com/overview/welcome/	Commercial, free option for academia and research	Agent-based modeling toolkit for simulating general concepts that unify macro- and micro-modeling
9	Simul8	2023	GUI-based programming	https://www.simul8.com/ https://web.archive.org/web/20230610043228/https://www.simul8.com/	Commercial license, some free options for academic use	Visual platform based on multimethod simulation models that support agent-based, discrete-event, and continuous methods both individually and in combinations

- **AnyLogic**—A modeling tool designed to support business simulations across a wide range of industries by offering dedicated libraries for domain-specific purposes. The software facilitates the multimethod modeling strategy and, as such, incorporates several modeling approaches, including agent-based, discrete-event, and system dynamics. All of these methods can be used separately or in combination with each other. With respect to the ABM, AnyLogic supports the development of various types of agents that can be encapsulated within one another or can form populations. The platform offers three types of spaces in which the agent can reside: continuous 3D space, grid-based discrete space, and geographic information system (GIS) space. Furthermore, AnyLogic supports the concepts of extended agent mobility (e.g., allowing specification of the time of agent movement) and offers various types of networks that can be used to define agent connections. It also implements inspection and statistic-collection tools that allow monitoring of the simulation flow;
- **ExtendSim**—A suite of simulation tools that enable continuous process modeling and discrete-event simulation. It offers different modeling approaches, including Monte Carlo, state/action and agent-based modeling. In the simulator, the agents are represented as ExtendSim blocks. They are perceived as individual decision-making entities, and their interaction is block-to-block-based. Furthermore, the platform supports removing/adding or moving agents from the model. ExtendSim consists of several modeling products. It includes (1) ExtendSim CP, which is a product designed for modeling and analysis of continuous, time-based operations; (2) ExtendSim DE, which is a tool facilitating the construction of a comprehensive message-based discrete-event architecture; and (3) ExtendSim Pro, which offers additional capabilities, such as the Discrete Rate Module;
- **FlexSim**—A tool that enables the modeling, simulation, prediction, and visualization of various business systems across multiple industries. The modeling is undertaken in 3D virtual environments. The agents are referred to as model objects. They are used

to generate events that are executed chronologically in a synchronized, discrete-event manner. The platform also facilitates agent (object) grouping with the support of different types of groups. Furthermore, FlexSim provides functionalities that can be used in agent learning; hence, it can serve as a tool to evaluate and implement the reinforcement learning algorithms. Additionally, the simulator also incorporates statistical distribution methods and random number generation (which account for the simulation variability). The software is also complemented by pre-made 3D models that facilitate visualization, as well as by a graphical interface;

- **FlexSim HC**—A specialized simulation software package designed for the healthcare industry. The tool is provided by the same company that developed FlexSim. It offers a variety of 3D models specifically tailored for modern medical facilities, providing a virtual environment for healthcare professionals to visualize and optimize their operations. From the conceptual perspective, it follows similar principles as FlexSim; however, the concept of 3D objects has some differences. For instance, FlexSim HC extends the idea of object groups into sub-groups and provides different object types (e.g., patients, not items). As such, the simulator allows emulating the flow of patients through the healthcare system, taking into account factors such as recoveries, patient outcomes, and time required for diagnoses;
- **GoldSim**—A platform that enables users to visually and dynamically model and simulate complex systems in a range of fields, including engineering, science, and business. It is based on a "visual spreadsheet" interface, which facilitates graphical creation, manipulation, and analysis of the data, equations, and models. For instance, it provides tools for probabilistic analysis, such as Monte Carlo simulations. The GoldSim simulations are built from modules that follow the top-down hierarchical approach. Furthermore, similarly to AnyLogic, GoldSim introduces different types of simulation approaches, including discrete-event simulations, agent-based simulations, and continuous simulations (as well as hybrid combinations of simulations). However, it does not provide explicit built-in mechanisms to facilitate them. Rather, it allows their exploitation through its general capabilities;
- Java Agent Construction Kit (**JACK**)—A Java-based development platform for creating MASs with the BDI approach. JACK agents can be described by means of their desires and can implement reasoning behaviors in response to both proactive (when the agent's knowledge changes) and reactive (in response to external events) triggers. JACK also provides generic templates that can help new developers in the creation of simple agent plans. As defining pre-compiled plans ensures system performance and predictability, JACK is suitable for time-sensitive and mission-critical applications. Additionally, JACK is complemented by the JACK Development Environment (JDE), which is a graphical tool that simplifies application development:
  (1) **C-BDI** is another framework distributed by the AOS company that developed JACK that aims to facilitate building human–machine teams based on the BDI approach. It aims at facilitating applications with distributed team reasoning, resilience, and scalability. The framework utilizes an explainable AI in its models, making the applications more efficient. The modeling is undertaken through a dedicated GUI that also assists in the visualization of prepared simulations. It is critical to stress that one of the key objectives behind the C-BDI is ethical AI support;
  (2) **CoJACK** is the third product distributed by the same company and is intended specifically for the modeling of human behaviors. It applies the principles of cognition theory inside the virtual actors, making it suitable for modeling individual and team behaviors. Its primary applications include simulations of armed engagements with hostile forces;
- **Simio**—A platform that offers an object-based modeling environment that enables the construction of 3D models directly from a top-down 2D view. The platform facilitates step-based model creation in which models are created incrementally. Similarly to previously described platforms (AnyLogic and GoldSim), Simio also combines discrete-

event, agent-based, and continuous simulation approaches. The agents developed in Simio are defined using behavioral patterns and can further incorporate physical attributes (e.g., speed, orientation). Such agent-based models can also be used in combination with discrete-event simulations;

- **Simudyne**—SDK with an agent-based approach for building models that mimic the real world's complexity. Simudyne's agents can interact with each other and react to environmental changes. The relationships between them are represented by networks composed of links, which can be constructed using specific built-in graph-generation algorithms. The platform also allows the modification of the network's links during runtime, which supports their dynamic evolution. Agents communicate through message exchange, which facilitates both unordered and ordered methods of sending messages. Additionally, it provides an extensive dashboard to visualize simulation outcomes;
- **Simul8**—A tool that offers support for simulations across various domains, including those involving continuous processes, discrete events, or agent-based models. Among its features, it offers a drag-and-drop interface that simplifies the creation of simulations, provides a scenario manager that allows testing diverse simulation ideas, and incorporates tools enabling exporting of the simulation's results to external files. Additionally, it is equipped with the capability to perform optimization through the integration of the "OptQuest" plugin. Moreover, an online version of the software, "Simul8 Online", is also available and offers the same feature set as the desktop version.

All platforms considered above can be used for the implementation of agent systems in various domains. In some cases, however, they may lack intuitive tools that would facilitate the efficient development of specific types of applications. Therefore, there is a need for platforms that are focused on particular research areas and, hence, incorporate features and approaches specifically relevant to the given field. In light of this, the next section of this review focuses entirely on agent platforms that can be used only for specific types of applications.

5. Special-Purpose Platforms

In many cases, domain-specific platforms address the need for agent-based simulation and modeling of various complex phenomena. Several domains, including healthcare, transportation, economics, and ecology, are heavily influenced by human behavior. Hence, traditional modeling techniques often fall short of capturing the complexities of such systems. Interesting examples include simulations of the evacuation of crowded spaces [118] and of traffic congestion [119]. In these cases, the behaviors of individuals are difficult to simulate as they depend on complex reasoning processes and can be subject to unpredictable changes. Agent-based simulations provide the capability to model individuals and simulate their interactions. As individuals are represented by agents, they exhibit autonomous behaviors, which allow a more realistic and detailed understanding of the system dynamics. Due to this, there is a need for dedicated frameworks that can facilitate the implementation of such agent-based models. The domain-specific platforms provide particular features that can be used specifically in the given area of implementation. These features may include, for instance, dedicated components that are associated with the environment of a particular type of application. In this section, as presented in Figure 1, the agent platforms are divided according to their implementation area. Hence, let us proceed to the review of the first group of agent platforms, which specifically address the modeling of social phenomena.

5.1. Cognitive, Social, and Affective Agent Platforms

Social simulations are one of the broadest areas of implementation of multi-agent systems. The platforms that are facilitating the development in that field are particularly used in the modeling of human behaviors. Therefore, in many cases, they incorporate elements of cognitive theory and the BDI approach. Some platforms provide features that are focused on simulating the relations between individuals, whereas others put more emphasis on the interactions between entire populations or societies. The main features of the platforms belonging to this category are summarized in Table 4, while more detailed descriptions follow.

Table 4. Cognitive, social, and affective platforms.

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
1	ACT-R	7.27.7	ACT-R	http://act-r.psy.cmu.edu/ https://web.archive.org/web/20230610043500/http://act-r.psy.cmu.edu/ http://acs.ist.psu.edu/papers/ritterTOip.pdf https://web.archive.org/web/20230611174853/https://acs.ist.psu.edu/papers/ritterTOip.pdf	https://www.gnu.org/licenses/old-licenses/lgpl-2.1.en.html https://web.archive.org/web/20230610044027/https://www.gnu.org/licenses/old-licenses/lgpl-2.1.en.html	Cognitive architecture and theory for simulating and understanding human cognition
2	Cormas	02/2023	Smaltalk	http://cormas.cirad.fr/indexeng.htm https://web.archive.org/web/20220901040250/http://cormas.cirad.fr/indexeng.htm https://github.com/cormas/cormas https://web.archive.org/web/20230610044156/https://github.com/cormas/cormas	https://github.com/cormas/cormas/blob/master/LICENSE https://web.archive.org/web/20230610044203/https://github.com/cormas/cormas/blob/master/LICENSE	ABM platform for simulating renewable resource management and societies' relationships and environments
3	DALI	2021.06	Prolog	https://github.com/AAAI-DISIM-UnivAQ/DALI https://web.archive.org/web/20230610044354/https://github.com/AAAI-DISIM-UnivAQ/DALI http://ceur-ws.org/Vol-1949/CILCpaper05.pdf https://web.archive.org/web/20230610044404/https://ceur-ws.org/Vol-1949/CILCpaper05.pdf	https://github.com/AAAI-DISIM-UnivAQ/DALI/blob/master/LICENSE https://web.archive.org/web/20230610044441/https://github.com/AAAI-DISIM-UnivAQ/DALI/blob/master/LICENSE	Prolog logic-based framework for defining agents and MASs with potential robotic applications
4	GOAL	2.2.0	Prolog	https://goalapl.atlassian.net/wiki/spaces/GOAL/overview?homepageId=32946 https://web.archive.org/web/20230610044514/https://goalapl.atlassian.net/wiki/spaces/GOAL/overview?homepageId=32946 https://bitbucket.org/goalhub/workspace/projects/GOAL	https://www.gnu.org/licenses/gpl-3.0.txt https://web.archive.org/web/20230610044547/https://www.gnu.org/licenses/gpl-3.0.txt	Agent programming language for cognitive agents that derive their choice of action from their beliefs and goals
5	GROWLab	0.9.7	Java	https://icr.ethz.ch/research/growlab/ https://web.archive.org/web/20230610044615/https://icr.ethz.ch/research/growlab/ https://icr.ethz.ch/research/growlab/publications/ https://web.archive.org/web/20230610044732/https://icr.ethz.ch/research/growlab/publications/	https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html https://web.archive.org/web/20230610044756/https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html	Software toolbox for agent-based simulation for complex social processes, especially conflict research

Table 4. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
6	JASON	3.2.0	AgentSpeak	http://jason.sourceforge.net/wp/ https://web.archive.org/web/20230610044824/https://jason.sourceforge.net/wp/ https://sourceforge.net/projects/jason/ https://web.archive.org/web/20230610044902/https://sourceforge.net/projects/jason/ https://github.com/jason-lang/jason https://web.archive.org/web/20230610044911/https://github.com/jason-lang/jason	https://github.com/jason-lang/jason/blob/master/LICENSE https://web.archive.org/web/20230610044928/https://github.com/jason-lang/jason/blob/master/LICENSE	Interpreter of an improved version of the AgentSpeak(L) language that adheres to the BDI approach
7	Neural MMO	2.0	Python	https://neuralmmo.github.io/_build/html/rst/landing.html https://web.archive.org/web/20230610045150/https://neuralmmo.github.io/_build/html/rst/landing.html https://github.com/NeuralMMO https://web.archive.org/web/20230610045221/https://github.com/NeuralMMO https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/file/44f683a84163b3523afe57c2e008bc8c-Paper-round1.pdf https://web.archive.org/web/20230610045211/https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/file/44f683a84163b3523afe57c2e008bc8c-Paper-round1.pdf	https://github.com/NeuralMMO/environment/blob/v1.6/LICENSE https://web.archive.org/web/20230610045231/https://github.com/NeuralMMO/environment/blob/1.6/LICENSE	Research platform that simulates populations of agents in procedurally generated virtual worlds
8	PAXelerate	1.1	Java	https://bauhausluftfahrt.github.io/PAXelerate/ https://web.archive.org/web/20230610045334/https://bauhausluftfahrt.github.io/PAXelerate/ https://github.com/BauhausLuftfahrt/PAXelerate https://web.archive.org/web/20230610045315/https://github.com/BauhausLuftfahrt/PAXelerate https://doi.org/10.3390/aerospace7120182 https://web.archive.org/web/20230611175209/https://www.mdpi.com/2226-4310/7/12/182	https://github.com/BauhausLuftfahrt/PAXelerate/blob/main/LICENSE https://web.archive.org/web/20230610045326/https://github.com/BauhausLuftfahrt/PAXelerate/blob/main/LICENSE	Two-dimensional (2D) ABM platform with aspects of cellular automaton (CA) used in simulations of aircraft cabin passenger flow

Table 4. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
9	Sim2APL	1.0.0	Java	https://bitbucket.org/goldenagents/sim2apl/src/master/ https://link.springer.com/chapter/10.1007/978-3-030-97457-2_1 https://web.archive.org/web/20230611175200/https://link.springer.com/chapter/10.1007/978-3-030-97457-2_1	https://bitbucket.org/goldenagents/sim2apl/src/master/LICENSE https://web.archive.org/web/20230611175634/https://bitbucket.org/goldenagents/sim2apl/raw/4e761bc59e0346ba54c65bcf589eaeef44b2bdf9/LICENSE	A 2APL Java library for step-based social simulations meant for reproducible, deterministic simulations with BDI agents
10	SOAR	9.6.1	Soar	https://soar.eecs.umich.edu/ https://web.archive.org/web/20230610045452/https://soar.eecs.umich.edu/ https://github.com/SoarGroup/Soar https://web.archive.org/web/20230610045521/https://github.com/SoarGroup/Soar https://arxiv.org/abs/2205.03854 https://web.archive.org/web/20230611175102/https://arxiv.org/abs/2205.03854	https://github.com/SoarGroup/Soar/blob/development/LICENSE.md https://web.archive.org/web/20230610045711/https://github.com/SoarGroup/Soar/blob/development/LICENSE.md	An architecture for simulating cognitive, intelligent agents that can be used across the full range of agent research problems
11	SOSIEL	2.4.6	C#	https://www.sosiel.org/ https://web.archive.org/web/20230610045804/https://www.sosiel.org/ https://github.com/SOSIEL/Algorithm-SOSIEL https://web.archive.org/web/20230610045814/https://github.com/SOSIEL/Algorithm-SOSIEL	https://github.com/SOSIEL/Algorithm-SOSIEL/blob/master/COPYING.LESSER https://web.archive.org/web/20230610045820/https://github.com/SOSIEL/Algorithm-SOSIEL/blob/master/COPYING.LESSER https://github.com/SOSIEL/Algorithm-SOSIEL/blob/master/COPYING https://web.archive.org/web/20230610045905/https://github.com/SOSIEL/Algorithm-SOSIEL/blob/master/COPYING	Multi-agent algorithm for simulating social contexts of agents that learn, interact, and make decisions in a complex environment

- **ACT-R**—A system that resembles a programming language but its underlying constructs reflect assumptions about human cognition taken from empirical findings of psychological experiments. The framework architecture is based on the modules that allow simulating cognitive processes (e.g., decision making, perception, and attention memory) and, as such, facilitate cognitive reasoning in agents. The agents rely on the knowledge base, which includes "facts", "concepts", and "rules" that dictate their behaviors. They are able to interact with the environment through perceptual interfaces. Furthermore, ACT-R provides metrics that enable the evaluation of the models' performance, including traditional cognitive psychology metrics, such as task completion time, accuracy, and neuroimaging. As such, the language facilitates comparisons of collected results with those derived from actual human subjects;
- **Cormas**—Agent-based modeling platform designed to emulate communal resource management and the influence that a population has on its surrounding natural

environment. The software is aimed at modeling the interactions among the different participants involved in managing sustainable natural resources. The platform enables the creation, examination, and analysis of simulated scenarios. Moreover, it allows for interaction with the simulation in real time by modifying agent behavior and resource usage. The implementation of customized models is facilitated by the use of generic classes offered by Smalltalk. Additionally, Cormas includes addons that enable the generation of source code from UML class diagrams and support R routines for remote control of simulations;

- **DALI—**A meta-interpreter built on top of Sicstus Prolog. It is a logic programming language that allows the usage of computational logic in the context of agent systems. The primary objective of the specification of this language is the identification and formalization of the basic patterns for proactivity, reactivity, autonomous "thinking", and "memory". The DALI agents are built upon the YARP communication toolkit, which enables them to exchange action commands with a platform. As a result of the integration of ServerDALI, the agents may also be placed on a server, which allows them to be accessed from an external environment;
- **GOAL—**A programming language designed for creating agents with cognitive capabilities. Specifically, it facilitates the programming at the "knowledge level". The information possessed by the agent is represented using KR semantics (e.g., OWL/prolog), and these semantics are also used in agent communication. GOAL facilitates dynamic changes in the beliefs and goals of an agent and allows for the structuring of their decision-making processes. The platform implements the blind commitment strategy, meaning that the agent's goals are discarded only upon their achievement. Additionally, GOAL supports agent training by implementing reinforcement learning algorithms;
- **GROWLab—**A Java-based software toolbox for modeling social phenomena in the field of geographic conflict research. Specifically, it is intended for applications that require the integration of actual GIS data. Its design considers the implementation of hierarchical agent structures. In particular, it provides notions of "layers" (i.e., groups of agents), "topologies" (i.e., agent interconnections), and "configurations" (i.e., agent hierarchies). Furthermore, the platform provides different types of spaces (e.g., grids, hexagonal spaces, geographic spaces). In order to assess the simulation results, GROWLab also offers tools for gathering statistics and model visualization;
- **JASON—**An interpreter for an extended version of the AgentSpeak programming language. In particular, it is characterized by speech act-based inter-agent communication. It is used for creating MASs based on the BDI architecture (e.g., ref [120] used JASON in DEVS simulation). As such, it provides concepts of perceptions, beliefs, desires, and intentions. Furthermore, by incorporating features from other frameworks, it facilitates the programming of multi-agent system organizations (by using Moise) and supports the distributivity of agents across the network (by using JADE);
- **Neural MMO—**An open-source research platform that allows for the simulation of populations of agents within virtual worlds generated using procedural methods. As such, the tool can facilitate foraging tasks, including survival, exploration, and combat, involving a large number of agents generated over several hours. The agents reside in auto-generated environments (initially at random locations) and perform the action in each simulation tick according to the observed game state. Furthermore, they are capable of interacting with each other (e.g., competing for food). In order to control the agents, Neural MMO uses the policies parameterized by neural networks;
- **PAXelerate—**A software that creates simulations of movements of passengers in airplane cabins. It aims to improve the layout of the boarding-process cabins. The system consists of two main parts: (1) a cabin editor that allows for changes to the cabin layout and (2) a simulator that runs the boarding simulation. The passengers are represented by computer-generated agents with randomly generated physical characteristics who find their seats using an A-Star path-finding algorithm. The simulation uses a grid

of nodes to represent the cabin area, and movement is based on cellular automaton principles. As such, PAXelerate can also be used to facilitate real-time simulations;

- **Sim2APL**—A library facilitating social simulations with intelligent agents. It is based on the 2APL architecture and enables agents to coordinate their actions using time intervals called "ticks". Each tick represents a complete thinking process for the agents, involving sensing, reasoning, and acting. Unlike the standard 2APL, Sim-2APL relies on agents generating action references instead of directly affecting the environment. After each tick, the actions are carried out in a predictable way, making it possible to create simulations that are consistent and predictable even with complex, intelligent agents. The library comes with pre-set options for starting, scheduling, and running agents. Agents can be created, modified, and removed using a central component called the Platform;
- **Soar**—A cognitive architecture designed to create intelligent agents. It aims to generate agents with the ability to mimic the complete range of cognitive capabilities of a human, such as learning, decision making, and adapting. Agents themselves are composed of fixed blocks provided by the architecture. They make the decisions based on (1) sensed data, (2) the contents of the working memory, and (3) long-term knowledge. The platform is still active and has often been used by AI researchers since 1983, mainly because of the native support for reinforcement learning for each agent;
- Self-Organizing Social and Inductive Evolutionary Learning (**SOSIEL**)—A multi-agent algorithm simulating social phenomena that considers large areas and long time spans. Each agent has its personalized knowledge, and sharing it is bounded by spatial criteria. Each agent can learn from the experiences of others and can adapt their practices based on that knowledge. The algorithm allows for the process of modeling the collective adaptation of the whole population after a single member has gained sufficient experience and starts sharing it.

Agent platforms for cognitive and social simulations allow the preparation of small and large agent-based models that can be used to understand the complexities of human behavior and social interactions. The social simulations can be additionally advanced by the incorporation of machine learning techniques [121]. As such, in recent years, it has become more popular to incorporate AI training into various agent-based applications. Hence, the next section focuses on agent platforms dedicated to supporting the implementation of learning agents.

5.2. Platforms for Learning Agents

The platforms considered in this section have been developed to address the needs of research on artificial intelligence. Mostly, they have been created to support simulations involving the training of AI-controlled agents. These agent platforms can be applied in domains such as game development or robotics. The main facts about the platforms are summarized in Table 5, followed by more detailed descriptions.

Table 5. Platforms for Learning Agents.

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
1	Brax	0.9.0	Python	https://github.com/google/brax https://web.archive.org/web/20230610050212/https://github.com/google/brax https://arxiv.org/pdf/2106.13281.pdf https://web.archive.org/web/20230610050220/https://arxiv.org/pdf/2106.13281.pdf	https://github.com/google/brax/blob/main/LICENSE https://web.archive.org/web/20230610050231/https://github.com/google/brax/blob/main/LICENSE	Physics engine containing algorithms for agent training used in robotics research

Table 5. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
2	Deepmind Garage	2020.06.3	Python	https://github.com/rlworkgroup/garage https://web.archive.org/web/20230610050323/https://github.com/rlworkgroup/garage	https://github.com/rlworkgroup/garage/blob/master/LICENSE https://web.archive.org/web/20230610050331/https://github.com/rlworkgroup/garage/blob/master/LICENSE	Toolkit that facilitates reinforcement learning algorithms and provides a standardized, reproducible environment
3	Deepmind Lab	December 2020	C, Lua	https://github.com/deepmind/lab https://web.archive.org/web/20230610050403/https://github.com/deepmind/lab https://arxiv.org/abs/1612.03801 https://web.archive.org/web/20230611175921/https://arxiv.org/abs/1612.03801	https://github.com/deepmind/lab/blob/master/LICENSE https://web.archive.org/web/20230610050643/https://github.com/deepmind/lab/blob/master/LICENSE	Three-dimensional (3D) game platform used for exploring learning agents that aims to serve as a testing environment in AI research
4	Gazebo	Garden	C++	https://gazebosim.org/home https://web.archive.org/web/20230610050420/https://gazebosim.org/home https://github.com/gazebosim/gz-sim https://web.archive.org/web/20230610050553/https://github.com/gazebosim/gz-sim	https://github.com/gazebosim/gz-sim/blob/ign-gazebo6/LICENSE https://web.archive.org/web/20230610050526/https://github.com/gazebosim/gz-sim/blob/ign-gazebo6/LICENSE	Cross-platform 3D robotics simulator that facilitates high-fidelity physics, rendering, and sensor models
5	Gymnasium	0.28.1	Python	https://gymnasium.farama.org/ https://web.archive.org/web/20230610050603/https://gymnasium.farama.org/ https://github.com/Farama-Foundation/Gymnasium https://web.archive.org/web/20230610050632/https://github.com/Farama-Foundation/Gymnasium	https://github.com/Farama-Foundation/Gymnasium/blob/main/LICENSE https://web.archive.org/web/20230610050727/https://github.com/Farama-Foundation/Gymnasium/blob/main/LICENSE	Toolkit for developing reinforcement learning algorithms that facilitate API for single-agent RL
6	Habitat	0.2.4	Python	https://aihabitat.org/ https://web.archive.org/web/20230610050741/https://aihabitat.org/ https://github.com/facebookresearch/habitat-sim https://web.archive.org/web/20230610050752/https://github.com/facebookresearch/habitat-sim https://github.com/facebookresearch/habitat-lab https://web.archive.org/web/20230610050841/https://github.com/facebookresearch/habitat-lab	https://github.com/facebookresearch/habitat-sim/blob/main/LICENSE https://web.archive.org/web/20230610050854/https://github.com/facebookresearch/habitat-sim/blob/main/LICENSE https://github.com/facebookresearch/habitat-lab/blob/main/LICENSE https://web.archive.org/web/20230610050901/https://github.com/facebookresearch/habitat-lab/blob/main/LICENSE	Simulation platform that enables training of embodied AI agents and offers a high-performance 3D simulator

Table 5. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
7	MAgent2	0.3.2	Python	https://magent2.farama.org/ https://web.archive.org/web/20230610051153/https://magent2.farama.org/ https://github.com/Farama-Foundation/MAgent2 https://web.archive.org/web/20230610051210/https://github.com/Farama-Foundation/MAgent2	https://github.com/Farama-Foundation/MAgent2/blob/main/LICENSE https://web.archive.org/web/20230610051207/https://github.com/Farama-Foundation/MAgent2/blob/main/LICENSE	Engine for generating high-performance multi-agent grid environments with many agents
8	Mava	0.1.3	Python	https://id-mava.readthedocs.io/en/latest/ https://web.archive.org/web/20230610051400/https://id-mava.readthedocs.io/en/latest/ https://github.com/instadeepai/Mava https://web.archive.org/web/20230610051407/https://github.com/instadeepai/Mava	https://github.com/instadeepai/Mava/blob/develop/LICENSE https://web.archive.org/web/20230610051453/https://github.com/instadeepai/Mava/blob/develop/LICENSE	Library for building multi-agent reinforcement learning systems using modular building blocks
9	MuJoCo	2.3.3	C, C++, C#, Python	https://mujoco.org/ https://web.archive.org/web/20230610051500/https://mujoco.org/ https://github.com/deepmind/mujoco https://web.archive.org/web/20230610051507/https://github.com/deepmind/mujoco	https://github.com/deepmind/mujoco/blob/main/LICENSE https://web.archive.org/web/20230610051554/https://github.com/deepmind/mujoco/blob/main/LICENSE	General-purpose physics engine with native 3D modeling environment that facilitates fast and reliable simulations
10	Open Spiel	1.2	C++, Python	https://github.com/deepmind/open_spiel https://web.archive.org/web/20230610051602/https://github.com/deepmind/open_spiel	https://github.com/deepmind/open_spiel/blob/master/LICENSE https://web.archive.org/web/20230610051633/https://github.com/deepmind/open_spiel/blob/master/LICENSE	Collection of environments and algorithms for research on general reinforcement learning and searching/planning in games
11	PySC2	4.0	Python	https://github.com/deepmind/pysc2 https://web.archive.org/web/20230610051659/https://github.com/deepmind/pysc2 https://arxiv.org/abs/1708.04782 https://web.archive.org/web/20230611180935/https://arxiv.org/abs/1708.04782	https://github.com/deepmind/pysc2/blob/master/LICENSE https://web.archive.org/web/20230610051715/https://github.com/deepmind/pysc2/blob/master/LICENSE	Reinforcement learning environment inspired by StarCraft II that involves multi-agent interactions
12	Unity ML-agents	20	C#, Python	https://github.com/Unity-Technologies/ml-agents https://web.archive.org/web/20230610051724/https://github.com/Unity-Technologies/ml-agents	https://github.com/Unity-Technologies/ml-agents/blob/main/LICENSE.md https://web.archive.org/web/20230610051812/https://github.com/Unity-Technologies/ml-agents/blob/main/LICENSE.md	Platform that facilitates the training of intelligent agents that is dedicated to game development and AI research

- **Brax**—A physics engine that aims to facilitate developments in the areas of robotics, human perception, materials science, and reinforcement learning, among others [122].

It supports both single-device simulations and distributed, parallelized ones. The engine was implemented using JAX and, as such, it can facilitate the environment's simulation of millions of "physics steps" per second and support fast agent training. The training is undertaken using already implemented algorithms provided in Brax (e.g., proximal-policy optimization, evolutionary strategy). In order to simulate the physics in the system, the platform incorporates the concepts of rigid bodies. Furthermore, it provides interfaces that allow for model visualization;

- **Deepmind Garage**—A toolkit for the development and assessment of reinforcement learning algorithms. It provides features for defining custom environments in which agents (robots) are represented by 2D observation points executing 2D actions (defined as velocities). The state of the agents is updated in a step-based manner. Furthermore, the platform provides a set of libraries that contain constructs that can be used in the implementation of agent training algorithms. Among other resources, it includes composable neural network models, replay buffers, and high-performance samplers. The platform also provides a graphical interface for defining experiments and experiment checkpointing tools;
- **Deepmind lab**—A platform that offers a set of tasks requiring agents to implement 3D navigation and puzzle solving. These tasks are created by adapting levels from Quake III Arena, a first-person shooter game, as well as custom levels designed by the community using the platform's built-in level editor. Its primary aim is to serve as a testing ground for artificial intelligence research, particularly in the field of deep reinforcement learning. Furthermore, utilizing Quake levels as the environment for simulations allows for comparisons of the performance of the implemented learning agent algorithms against the behavior of the significant Quake player base;
- **Gazebo**—A 3D dynamic software package that allows for the simulation of populations of robots in both indoor and outdoor environments. It is based on a robust physics engine (DART) with support for kinematic and dynamic applications. Furthermore, the platform facilitates realistic simulations through a set of sensor and noise models, as well as the incorporation of 3D rendering techniques. The simulations implemented in Gazebo are time step-based; therefore, the platform can also be used for real-time simulations. In particular, it is commonly used for rapid testing of robotics algorithms and performing regression testing in realistic scenarios;
- **Gymnasium**—Open-source Python library for creating and analyzing reinforcement learning algorithms. It achieves this goal by providing a standardized API that enables communication between learning algorithms and environments. The actions are executed on the environment in a step-based manner, where "steps" refer to the agent observations (i.e., rewards resulting from actions). Besides facilitating the creation of custom environments, the platform also provides a predefined collection that can be used in the implementation of models in domains including Atari games, robotics simulations, and control problems;
- **Habitat**—Simulation toolkit that facilitates the development and training of embodied AI agents. It allows visualizing agents in photorealistic 3D environments. The toolkit consists of two main components: Habitat-Sim and Habitat-Lab. Habitat-Sim is a 3D high-performance simulator with, among other assets, enabled physics, rigid-body structures, CAD models of spaces, and configurable sensors. Habitat-Lab facilitates embodied AI tasks; specifically, the configuration and training of agents;
- **MAgent2**—A research tool designed for multi-agent reinforcement learning (MARL) with the goal of scaling up to a large number of agents. As such, Magent2 provides implementations of baseline algorithms, including parameter-sharing DQN, DRQN, and a2c, which are used in agent learning. The platform allows for the development of customized agent behaviors executable in a decentralized manner. The communication and collaboration between agent entities are undertaken using global observations of the environment and of the states of other agents. Similarly to the previous platforms, the state of the system is defined in a step-based manner. Although the project was

originally discontinued by its authors, it has since been taken up by the Farama Foundation and is now regularly updated;

- **Mava—**A comprehensive framework for creating multi-agent reinforcement learning (MARL) that offers modular abstractions. It follows the executor–trainer paradigm inspired by the actor–learner concept in distributed single-agent reinforcement learning. Mava facilitates the interaction between systems and the environment. It supports distributed system training and provides various components and architectures for system creation. Furthermore, it offers baseline system implementations and demonstrates its functionality by featuring modularized, modifiable components. Additionally, Mava incorporates Launchpad, which enables the seamless launch of distributed multi-agent MARL systems;
- **MuJoCo—**A physics engine designed to support research and development applications in the robotics and machine learning fields. It is able to handle high-dimensional states and action spaces and supports multiple types of controllers (e.g., position-based, velocity-based, torque-based). The physical interactions that can be modeled in the system include contacts, friction, and joint constraints. Furthermore, its engine combines two techniques: generalized coordinates (for representing the state of the system) and optimization-based contact dynamics (for modeling physical interactions). Moreover, MuJoCo features its own modeling language, MJCF, which allows users to create and manipulate physical models. Additionally, it includes a native 3D visualizer based on open GL that supports real-time interactions with the models;
- **OpenSpiel—**A research platform for developing and studying games and algorithms in the domain of reinforcement learning and game theory. It offers several simulations (e.g., n-player zero-sum, cooperative and general-sum, and perfect and imperfect information games) and provides tools for agent-based modeling and analysis. It also supports interfaces for developing both single-agent and multi-agent reinforcement learning models. In this context, it provides built-in algorithms, such as Monte Carlo tree search (MCTS) and counterfactual regret minimization (CFR). It also enables developers to define custom games and extend the library with new game implementations. Additionally, it offers a set of evaluation tools and benchmarking capabilities;
- **PySC2—**An environment intended for training agents based on Starcraft II that provides a comprehensive simulation of the game. It implements algorithms facilitating training agents using reinforcement learning techniques. The platform supports real-time game-state observation and reward access, enabling an informed decision-making process in agents. Furthermore, it provides support for two types of agents: scripted and random. These agents are able to interact with each other and the environment using the game's API, and as such, they can perform actions, including movement or combat. Additionally, the visualization of prepared simulations is achieved through integration with the StarCraft II game client, which provides tools to render game states, agent actions, and maps;
- **Unity ML-Agents—**A game-based platform for developing intelligent agents. It provides tools for creating virtual environments for simulations in 2D, 3D, VR, and AR formats, as well as facilitating the handling of sensory input, such as visual and auditory data. The platform supports both single-agent and multi-agent training, including distributed training in which multiple agents are trained simultaneously. It provides three types of agent behaviors: (1) learning (when the agent is about to be trained), (2) heuristic (hard-coded rules), and (3) inference (training with neural networks). Additionally, it contains built-in tools (Demonstration Recorder) to record the agent behaviors, which allows for the monitoring of the statistics of the models.

Platforms for learning agents provide comprehensive sets of built-in features ranging from simple machine learning algorithms to more advanced learning techniques, such as reinforcement learning or neural network training. As a result, such platforms simplify the implementation of the learning agents since developers can rely on built-in algorithms and focus primarily on agents' modeling. Due to this, researchers without advanced programming skills can also use such platforms for their research projects that incorporate agents' training. This kind of research may be related to another field that uses agent systems that has not yet been discussed; namely, the modeling of ecosystems and environmental resources. Hence, let us now focus on the platforms that are specifically intended for that domain.

5.3. Platforms for Modeling and Simulating Environments and Ecosystems

Simulations of natural phenomena are another area of research that benefit greatly from agent-based models. Examples include models of specific ecosystems, biological simulations, land-use simulations, and simulations relating to the exploitation of environmental resources. Among more recent fields that can be included in this category, one can find simulations of grid micro-environments [123]. As such, the agent platforms from this domain are, in most cases, oriented towards large-scale simulations with the support of spatial modeling. Individual features of each of the selected platforms are described in Table 6. Then, each platform's characteristics are presented in greater detail.

Table 6. Environments and ecosystems modeling platforms.

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
1	Alchemist	15.16.2	Java	http://alchemistsimulator.github.io/ https://web.archive.org/web/20230610053639/http://alchemistsimulator.github.io/ https://github.com/AlchemistSimulator/Alchemist https://web.archive.org/web/20230610053647/https://github.com/AlchemistSimulator/Alchemist	http://alchemistsimulator.github.io/license/index.html https://web.archive.org/web/20230610053715/http://alchemistsimulator.github.io/license/index.html	Simulator for pervasive computing and distributed systems inspired by stochastic chemistry
2	BioDynaMo	1.05.63	C++	https://www.biodynamo.org/ https://web.archive.org/web/20230610061401/https://www.biodynamo.org/ https://github.com/BioDynaMo/biodynamo https://web.archive.org/web/20230610053740/https://github.com/BioDynaMo/biodynamo https://doi.org/10.1093/bioinformatics/btab649 https://web.archive.org/web/20230611182311/https://academic.oup.com/bioinformatics/article/38/2/453/6371176	https://github.com/BioDynaMo/biodynamo/blob/master/LICENSE https://web.archive.org/web/20230610053922/https://github.com/BioDynaMo/biodynamo/blob/master/LICENSE	Agent-based simulation platform used for 3D biological simulations with modular and high-performance engine
3	FAME	Circinus	Java, Python	https://joss.theoj.org/papers/10.21105/joss.05087.pdf https://web.archive.org/web/20230610053926/https://joss.theoj.org/papers/10.21105/joss.05087.pdf https://gitlab.com/fame-framework https://web.archive.org/web/20230610053935/https://gitlab.com/fame-framework	https://gitlab.com/fame-framework/wiki/-/blob/master/LICENCE https://web.archive.org/web/20230610054054/https://gitlab.com/fame-framework/wiki/-/blob/master/LICENCE	Toolkit intended for energy system simulations that supports simulations on both PCs and HPC clusters

Table 6. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
4	ForestSim	1.0.2	Java	https://github.com/forestsim-mtu/forestsim https://web.archive.org/web/20230610054104/https://github.com/forestsim-mtu/forestsim https://www.sciencedirect.com/science/article/pii/S2352711018302310 https://web.archive.org/web/20230611182112/https://www.sciencedirect.com/science/article/pii/S2352711018302310	https://github.com/forestsim-mtu/forestsim/blob/master/LICENSE https://web.archive.org/web/20230610054304/https://github.com/forestsim-mtu/forestsim/blob/master/LICENSE	ABM used in research on policy for and sustainability of woody biomass-based biofuels and bioenergy options
4	Framsticks	5.0rc25	C++	http://www.framsticks.com/ https://web.archive.org/web/20230610054356/http://www.framsticks.com/	http://www.framsticks.com/a/doc_license.html https://web.archive.org/web/20230610054416/http://www.framsticks.com/a/doc_license.html	Three-dimensional (3D) artificial life simulator dedicated to modeling the mechanical structures and control systems of individuals
5	HexSim	4.0.20	Modeling achieved with GUI, engine written in C++ and GUI in C#	https://www.hexsim.net/home https://web.archive.org/web/20230610054420/https://www.hexsim.net/home	Free, closed-source	Spatially explicit, individual-based simulator for modeling multispecies plant and animal populations with dynamic life history traits of individuals
6	LANDIS-II	7.0	C#	https://www.landis-ii.org/home https://web.archive.org/web/20230610054514/https://www.landis-ii.org/home https://github.com/LANDIS-II-Foundation https://web.archive.org/web/20230610054525/https://github.com/LANDIS-II-Foundation	https://github.com/LANDIS-II-Foundation/Core-Model-v6/blob/master/LICENSE.txt https://web.archive.org/web/20230610054533/https://github.com/LANDIS-II-Foundation/Core-Model-v6/blob/master/LICENSE.txt	Forest landscape model with multi-century timescales and spatial scales that can span millions of hectares
7	lemlab	1.0	Python	https://github.com/tum-ewk/lemlab https://web.archive.org/web/20230610054616/https://github.com/tum-ewk/lemlab https://lemlab.readthedocs.io/en/latest/index.html https://web.archive.org/web/20230610054624/https://lemlab.readthedocs.io/en/latest/index.html	https://github.com/tum-ewk/lemlab/blob/master/LICENSE https://web.archive.org/web/20230610054631/https://github.com/tum-ewk/lemlab/blob/master/LICENSE	ABM platform for local energy market modeling that can facilitate both simulation and real-time simulation models
8	Osmose	4.3.3	R	https://osmose-model.org/ https://web.archive.org/web/20230610054716/https://osmose-model.org/ https://github.com/osmose-model/osmose https://web.archive.org/web/20230610054717/https://github.com/osmose-model/osmose	http://www.cecill.info/licences/Licence_CeCILL_V2-en.txt https://web.archive.org/web/20230610054728/http://www.cecill.info/licences/Licence_CeCILL_V2-en.txt	Multispecies and individual-based modeling framework dedicated to fish species predator–prey simulations

Table 6. *Cont.*

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
9	ReMobidyc	preview-8	Smaltalk	https://github.com/ReMobidyc/ReMobidyc https://web.archive.org/web/20230610054820/https://github.com/ReMobidyc/ReMobidyc	https://github.com/ReMobidyc/ReMobidyc/blob/main/LICENSE https://web.archive.org/web/20230610054807/https://github.com/ReMobidyc/ReMobidyc/blob/main/LICENSE	Redesigned Mobidyc Web-based ABM platform for modeling population dynamics and ecotoxicology
10	Simona	2.1.0	Scala	https://simona.ie3.e-technik.tu-dortmund.de/ https://web.archive.org/web/20230610054821/https://simona.ie3.e-technik.tu-dortmund.de/ https://github.com/ie3-institute/simona https://web.archive.org/web/20230610054904/https://github.com/ie3-institute/simona	https://github.com/ie3-institute/simona/blob/dev/LICENSE https://web.archive.org/web/20230610054910/https://github.com/ie3-institute/simona/blob/dev/LICENSE	Agent-based simulation platform for power system simulations specifically involving large-scale electricity grids
11	TerraME	2.0.1	Lua	http://www.terrame.org/doku.php?id=start https://web.archive.org/web/20230610054919/http://www.terrame.org/doku.php?id=start https://github.com/TerraME/terrame https://web.archive.org/web/20230610055016/https://github.com/TerraME/terrame	https://github.com/TerraME/terrame/blob/master/LICENSE https://web.archive.org/web/20230610055009/https://github.com/TerraME/terrame/blob/master/LICENSE	Two-dimensional (2D) spatial dynamical simulator for ABM and network models that uses cellular/hybrid automata and anisotropic spaces

- **Alchemist**—A simulation framework that incorporates chemistry terminology within an agent-based system. Initially, the platform was dedicated strictly to chemistry-oriented multi-compartment stochastic simulations [124]. However, the chemistry-inspired concepts of the Alchemist's meta-model are now treated in an "abstract" manner. For instance, the model defines "nodes", "reactions", "molecules", and "concentrations". In this context, the "molecules" can be interpreted as identifiers and the "concentrations" as pieces of data. The Alchemist's meta-model also includes "incarnations", which are concrete instances of a given type. As such, these concepts allow for the definition of cognitive agents, which are created by adding a particular type to given node properties. Additionally, in order to streamline the simulation, the platform provides an incarnation-agnostic simulation engine with a customizable GUI;
- **BioDynaMo**—A simulation framework designed for multi-scale agent-based modeling of biological systems. Its design principles emphasize two main aspects: (1) a high-performance engine and (2) a modular component design. The agents are used to represent individual cells that reside in a 3D environment and are capable of dividing (creation of new agents), moving, and interacting with each other. The platform also supports the creation of agents in batches. From the technical perspective, the BioDynaMo simulation engine is fully parallelized, allowing it to be run on multi-core CPUs. As such, the platform is compatible with both standard computers and HPCs. By enabling GPU acceleration for faster processing, BioDynaMo can be used to simulate large-scale models;
- **FAME**—Toolkit that facilitates the development of agent-based models and simulations for energy systems. It consists of several modules, including FAME-Core, FAME-Prepare, FAME-Io, and FAME-MPI. The agent-based models are being developed using FAME-Core. It aims to provide built-in functionalities to simplify ABM

development for programmers without specialized knowledge. The FAME agents are able to interact with the environment, process the input configuration files, and produce the output data. The interaction between multiple agents is coordinated by the concept of contracts. The scheme of the defined models is prepared by the FAME-Prepare component, whereas the input and output data of the simulation are handled by FAME-Io. Additionally, FAME provides a FAME-MPI package for running parallelized applications on clusters using MPI;

- **ForestSim—**An agent-based model for researching policy and sustainability issues related to woody biomass-based biofuels and bioenergy options. The platform is Java-based and relies on the MASON toolkit for simulation management. It is designed to be modifiable for the purpose of studying more specific research areas. Among its features, it integrates techniques from biomass estimation, ABM, sustainability assessment, and forest-growth modeling. By simulating the harvest activities of decentralized private forest owners, it can be used to aid research aiming to determine the impact of changing forest management policies on locally derived sustainability indicators. Based on the information obtained from the platform's authors, ForestSim is currently in a stable state, and there are no plans to enhance it with new features;
- **Framesticks—**A system for modeling artificial life and observing the effects of aimless evolutionary mechanisms. It allows for separate simulations of the physical and neural components of a creature, taking into consideration genotype–phenotype mappings. The physical simulation consists of representing the creature's body as a finite number of points and considering the effects of forces applied on it by other points, as well as outside physical forces (i.e., gravity, friction, elasticity). The neural part of the system represents a creature's brain as a neural network. Moreover, Framsticks provides two sets of neurons: actuators responsible for the orientation of the creature's body and receptors providing information about the simulated environment. Besides that, users can define their own neuron types using editing scripts included in the platform. The system also includes a GUI for 3D simulation visualization;
- **HexSim—**A computer application specifically designed for constructing plant and animal population models in the field of computational biology research. Users can define the structure, complexity, and data requirements of their models within the HexSim platform. It leverages spatial data to capture aspects such as landscape structure, habitat quality, and the distribution of stressors. Moreover, the platform allows users to define the animal model structure by creating a customized life cycle. The life cycle serves as the driving force behind the model's processing and data requirements. Users can select from a list of life-history events, including survival, reproduction, movement, resource acquisition, and species interactions. This makes it useful for studying the cumulative impacts on wildlife populations and plants resulting from the interaction of multiple natural and human-induced disturbances;
- **LANDIS-II—**A modeling framework for simulating and studying the long-term ecological processes in forests. It represents trees and shrubs as individual agents that model their growth, mortality, and regeneration. The communication is undertaken through the effects of the agents' actions observable in the environment. As such, the platform enables the modeling of species interactions, including competition and facilitation. Furthermore, in the environment, the simulator incorporates spatially explicit factors (such as climate change) and disturbances (e.g., fire, insect outbreaks, harvesting). Additionally, it is worth pointing out that LANDIS-II models are customizable and, as a result, the platform has gathered an active community that has contributed significantly to the development of several extensions;
- **lemlab—**An agent-based modeling platform designed to facilitate the development and testing of local energy market (LEM) models. The platform is based on a modular design that incorporates a generic LEM architecture, which allows for the adaptation of the system to meet the research needs of the user. Lemlab includes integrated time-series data for simulating agents as several types of both power producers

and consumers, such as household loads, photovoltaic systems, wind turbines, heat pumps, and electric vehicles. It offers real-time capabilities for model development, along with an analysis toolbox for result testing. Furthermore, the platform also supports a database-agnostic approach, which facilitates the integration of multiple database technologies;

- **Osmose**—An R package for simulation of aquatic wildlife that can represent multiple different species in one environment. Specifically, the platform facilitates the implementation of individual-based and spatially explicit models. Its core concept is based on the co-occurrence of predators and prey. Each specimen (agent) represented in the simulation is characterized by a distinct set of parameters, such as weight, age, and size. Moreover, they undergo life cycle-related changes, including aging and reproduction. As such, the simulation is also capable of modeling predation between species, starvation in cases of insufficient resources, and introduction of the human element with fishing exploitation;
- **ReMobidyc**—A Web-based ABM platform used in individual-based modeling of population dynamics and ecotoxicology. From the theoretical perspective, it is based on the design principles of Mobidyc, but from the technical side, it utilizes Pharo instead of VisualWorks. In its design principles, the platform aims to facilitate the verification of models, reproduction of simulations, and tracing of the simulation process (by enabling the users to interact with agents at runtime). Therefore, ReMobidyc offers persistent storage (e.g., RDBs, which can be used in storing the simulation steps), a random number generator (which can support simulation reproducibility), and a Web-based UI (to make the systems shareable in the community);
- **Simona**—An agent-based discrete-event modeling tool dedicated to distributed grid simulations. It is primarily used for modeling power grids, their analysis, and planning. The grids are designed using input data that must adhere to the dedicated PowerSystemDataModel format. As the platform employs the individuality approach, it focuses on the behavior of individual assets, which are implemented as agents, rather than the entire grid. Additionally, it supports storing of the output of the simulation in specific output files that are configurable by the user;
- **TerraME**—A programming environment designed to simulate the impact of human-related and natural phenomena across anisotropic regions. It enables the implementation of three distinct modeling paradigms; namely, agent-based models, cellular automata, and network models. The simulations executed in TerraME are based on the events realized in discrete time steps. Furthermore, to execute scenarios, the platform can be integrated with GIS data through a customized TerraLib interface. Additionally, TerraMe includes a bundled 2D graphical user interface that focuses on capturing environmental variations within a grid.

Considering the platforms described in this section, it is evident that community support is an important factor in the development of open-source agent platforms. Platforms such as LANDIS-II are mostly maintained by active developers' communities. In terms of environment modeling (and modeling in general), this kind of approach is critical since it allows the community to contribute to the development of various models that can then be utilized to perform very specific types of simulations. Let us now complete the section on domain-specific platforms with a summary of platforms that are related to transport simulations.

5.4. Platforms for Transport-Related Simulations

In transport simulations, agent-based modeling is often used to represent individual vehicles or pedestrians as agents characterized by their own decision-making processes. These agents are typically simulated using agent platforms, which provide the underlying software infrastructure (often including runtime engines or visualization tools) for building and executing agent-based models. The platforms from this section are commonly used for traffic simulations in autonomous driving scenarios or in agent-based racing simulations.

They are summarized in Table 7. Following this, the characteristics of each of the platforms are described in more detail.

Table 7. Transport modeling platforms.

No.	Name	Version	Programming Language	Website, Documentation, and Projects	License	Description
1	AGADE Traffic	0.1.0	Java	https://agade.de/ https://web.archive.org/web/20230610055027/https://agade.de/ https://github.com/KITE-Cloud/AGADE-TRAFFIC https://web.archive.org/web/20230610055113/https://github.com/KITE-Cloud/AGADE-TRAFFIC	https://github.com/KITE-Cloud/AGADE-TRAFFIC/blob/master/LICENSE https://web.archive.org/web/20230610055121/https://github.com/KITE-Cloud/AGADE-TRAFFIC/blob/master/LICENSE	Traffic simulator that incorporates OWL and SWRL semantics and JADEX BDI agents
2	Carla	0.9.14	Python	https://carla.org/ https://web.archive.org/web/20230610055125/https://carla.org/ https://github.com/carla-simulator/carla https://web.archive.org/web/20230610055211/https://github.com/carla-simulator/carla	https://github.com/carla-simulator/carla/blob/master/LICENSE https://web.archive.org/web/20230610055216/https://github.com/carla-simulator/carla/blob/master/LICENSE	Three-dimensional (3D) autonomous driving simulator based on the Unreal Engine that serves as a tool in autonomous driving R&D
3	MATSim	14.0	Java	https://www.matsim.org/ https://web.archive.org/web/20230610055227/https://www.matsim.org/ https://github.com/matsim-org/matsim-libs https://web.archive.org/web/20230610055344/https://github.com/matsim-org/matsim-libs	https://www.gnu.org/licenses/gpl-3.0.html https://web.archive.org/web/20230610035641/https://www.gnu.org/licenses/gpl-3.0.html	Large-scale agent-based transport simulation for analysis of traffic and congestion patterns
4	Microsoft AirSim	1.8.1	C++, Java, C#, Python	https://github.com/microsoft/AirSim https://web.archive.org/web/20230610055325/https://github.com/microsoft/AirSim https://microsoft.github.io/AirSim/ https://web.archive.org/web/20230610055403/https://microsoft.github.io/AirSim/	https://github.com/microsoft/AirSim/blob/main/LICENSE https://web.archive.org/web/20230610055417/https://github.com/microsoft/AirSim/blob/main/LICENSE	Simulator for autonomous vehicles built on Unreal Engine and the Unity engine
5	Torcs	1.3.7	C/C++	http://torcs.sourceforge.net/index.php https://web.archive.org/web/20230610055500/https://torcs.sourceforge.net/index.php https://sourceforge.net/projects/torcs/ https://web.archive.org/web/20230610055512/https://sourceforge.net/projects/torcs/	https://www.gnu.org/licenses/old-licenses/gpl-2.0.html https://web.archive.org/web/20230610054616/https://www.gnu.org/licenses/old-licenses/gpl-2.0.html	Car racing real-time simulation with realistic 3D graphics used in AI racing games and research

- **AGADE Traffic**—An agent-based platform for traffic simulation. It utilizes NetLogo agents to model traffic participants and provides a graphical user interface for their visualization. The platform allows the specification of routes for individuals by defining their origins and destinations. Furthermore, AGADE enables the definition of cost and pricing schemes, which facilitates the comparison of routing and social optimization effects. The tool's architecture allows it to run in a distributed environment using the mechanisms of the graph database. Additionally, AGADE facilitates simulations of real-world scenarios by acquiring geographic information from OpenStreetMap;

- **Carla**—A platform for autonomous driving simulations. It includes built-in protocols and components for mobility modeling (e.g., city structures and vehicles). It enables control of static and dynamic actors used to represent vehicles. Furthermore, it provides sensor suites and implementation of environmental conditions. The platform consists of two main modules. The Carla simulator handles the majority of the workload and is responsible for rendering and managing the actors and sensors in the simulated environment. The Carla Python API provides an interface that enables control of vehicles, attaching of sensors to them, and collection of generated data;
- **MATSim**—An agent-based platform that provides a set of methods for running and implementing large-scale mobility simulations. It comprises "modules" (e.g., modules with configuration options) that can be used independently or in combination with each other. Moreover, it enables the definition of custom module implementations in order to test research-specific features. The MATSim simulations are run in a co-evolutionary manner, where agents are executed iteratively and compete for the space–time slots with other agents. A single iteration ends with the assessment of the agents' experiences. Furthermore, all of the events executed in the simulation are captured for analysis. MATSim also provides the dedicated "Via" visualization software that enables the display of the simulation results;
- **Microsoft AirSim**—A simulator for ground vehicles and aircraft that has versions using both the Unity engine and Unreal Engine. The environment is open-source and cross-platform and supports both software-in-the-loop simulation (with flight controllers) and hardware-in-loop simulation (for physically and visually realistic simulations). It defines vehicles as autonomous robots composed of shapes and sets of points, mapping them to corresponding forces. These robots are simulated in an environment that incorporates physics phenomena, such as gravity or air density. AirSim also provides multiple types of sensors, such as GPS, barometers, distance sensors, and Lidar. Moreover, it implements baseline learning algorithms (e.g., reinforcement learning), making it possible to build simulations in the AI research domain;
- The Open Racing Car Simulator (**Torcs**)—A portable, multi-platform game that can be used for both entertainment and research purposes. It incorporates an advanced physical model that allows for realistic car racing scenarios and offers a range of components, including racing tracks, opponents, and cars. Furthermore, the simulator features, among other elements, a damage model, collisions, and aerodynamics. In addition to being a car racing game, Torcs can be utilized as a platform for testing autonomous driving algorithms and, hence, can be used in the development of AI technology in the field of racing.

These traffic-related agent-based platforms complete the review of active agent platforms. As can be seen, there are numerous platforms that are under continuous development (i.e., all the platforms described above) and can assist scientists (and businesses) in the implementation of agent-based systems in a variety of application domains.

6. Concluding Remarks

Agent systems provide a bottom-up approach to addressing complex tasks. Agents also provide a means of modeling and simulating phenomena that are difficult to model and understand via standard analytical methods. Although the main applications of agent systems pertain to computer science and are often related to artificial intelligence, there are increasingly numerous uses of agents in areas such as the life sciences, ecological sciences, and social sciences. In response to these interests, numerous platforms have been developed as general or application-focused tools. Over time, the field of agent platforms has evolved, with new platforms being developed and others being abandoned.

This contribution provides a comprehensive review of the currently available (working and ready-to-use) agent platforms, highlighting their diversity across various domains. Its main objective is to assist researchers in assessing the characteristics of the platforms and selecting the most suitable ones for their scientific needs. As such, the presented work is supplemented by a dedicated website that contains an extended list of available (and past) agent platforms.

The review also provides the means to identify the current research trends in agent-based systems. Based on the platforms presented in this contribution, it is notable that agent systems are widely used in simulations, particularly concerning environmental and social phenomena. Additionally, the analysis of the recently developed platforms (i.e., LemLab and Simona), which are specifically dedicated to the simulation of power grids and electricity markets, indicates a strong research trend towards the incorporation of agents into the modeling and/or implementation of power system control.

On the other hand, this contribution also outlines the agent platform domains that have not yet been sufficiently covered. Examples include the lack of industrial-grade platforms (other than JADE and JACK) supported by companies that have greater access to financial and technological resources. In particular, the majority of the presented frameworks have been implemented by small research teams or individual creators, who, in some cases, may lack the long-term resources that would support the further development of their platforms. Consequently, there are many platforms for which development is unsteady and prone to abandonment over time.

Moreover, another gap in the agent platform domain concerns coordination and cloud-based agent platforms. The authors were able to find only a few available and maintained frameworks (i.e., TuCSoN, cloneMap). Here, it is noticeable that, in the case of platforms for agent coordination, the majority of existing examples have been proposed as mere proofs-of-concept, and the actual code base is not provided or was developed years ago and is no longer maintained. Moreover, in the case of cloud-based agent platforms, besides cloneMap, no other frameworks that encapsulate the agents in cloud clusters were found. The lack of platforms in these fields may arise from the complexity of the implementation of such systems' principles.

Particularly in terms of the agent platforms used to facilitate agent coordination and collaboration, aspects regarding the coordination and synchronization of agents represent ongoing research areas in the agent system domain. Since agents operate as autonomous entities with asynchronous behavior, designing protocols ensuring reliable and optimal communication mechanisms for agent cooperation and negotiation is challenging. This task is closely associated with another concern; namely, the scalability of agent-based systems, especially in the case of large-scale simulations. Even though some platforms support scalable computing, there still remains a need for easily accessible and scalable platforms that do not require specialized hardware (e.g., clusters) or advanced technical knowledge.

While the agent platforms cover a broad range of domains, allowing researchers from different fields to use agent-based systems in their works, several challenges and potential areas of improvement still remain. For instance, in terms of interoperability, most of the currently available platforms do not support communication between agents developed using different frameworks. There are only a few platforms (e.g., JADE) that adhere to standard protocols, such as FIPA, and, hence, enable interaction between agents residing in different environments. In this context, one area of improvement would involve incorporating common standards and protocols into the existing agent platforms. On the other hand, this problem could also be minimized by developing interoperability tools that would serve as middleware and translation layers for agent communication. Therefore, addressing these concerns could be a potential future direction for the agent platform domain that could streamline the implementation of agent-based systems across a wider range of applications.

Author Contributions: The authors participated equally in the preparation of all aspects of this contribution. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The contribution is supplemented by a dedicated WWW site that extends the material presented in the text with additional information regarding "past abandoned" and "likely abandoned" agent platforms. The website can be found at: https://www.ibspan.waw.pl//~paprzyck/mp/cvr/research/agent_platforms_site/agent_platforms.html (accessed on 3 June 2023). All suggestions regarding how the website can be further improved/corrected/extended with missing information are highly appreciated and should be sent to marcin.paprzycki@ibspan.waw.pl.

Acknowledgments: The authors would like to acknowledge and thank Costin Bădică, Lars Braubach, Giacomo Cabri, Stéphane Galland, P. Andrea Omicini, Piotr Pałka, and Franco Zambonelli for their support in improving this contribution.

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

References

1. Bădică, C.; Budimac, Z.; Burkhard, H.D.; Ivanovic, M. Software Agents: Languages, Tools, Platforms. *Comput. Sci. Inf. Syst.* **2011**, *8*, 255–298. [CrossRef]
2. Savaglio, C.; Ganzha, M.; Paprzycki, M.; Bădică, C.; Ivanović, M.; Fortino, G. Agent-based Internet of Things: State-of-the-art and research challenges. *Future Gener. Comput. Syst.* **2020**, *102*, 1038–1053. [CrossRef]
3. Weiss, G. *Multiagent Systems*, 2nd ed.; Intelligent Robotics and Autonomous Agents Series; MIT Press: Cambridge, MA, USA, 2013.
4. Wooldridge, M. *An Introduction to MultiAgent Systems*, 2nd ed.; Wiley: Hoboken, NJ, USA, 2009.
5. Pal, C.; Leon, F.; Paprzycki, M.; Ganzha, M. A Review of Platforms for the Development of Agent Systems. *arXiv* **2020**, arXiv:2007.08961. https://doi.org/10.48550/arXiv.2007.08961.
6. Davis, R. *Report on the Workshop on Distributed AI*; MIT Artificial Intelligence Laboratory: Cambridge, MA, USA, 1980.
7. Axelrod, R. *The Evolution of Cooperation*; Basic Books: New York, NY, USA, 1984.
8. Agha, G.; Hewitt, C. Concurrent Programming Using Actors: Exploiting Large-Scale Parallelism. *Readings Distrib. Artif. Intell.* **1988**, *24*, 398–407. [CrossRef]
9. Ericsson Computer Science Laboratory Erlang. Available online: https://web.archive.org/web/20230610060832/https://www.erlang.org/ (accessed on 12 April 2023).
10. Lightbend, Inc. Akka. Available online: https://web.archive.org/web/20230606075845/https://akka.io/ (accessed on 12 April 2023).
11. Asynkron AB Proto.Actor. Available online: https://web.archive.org/web/20230610060753/https://proto.actor/ (accessed on 12 April 2023).
12. Brooks, R. A robust layered control system for a mobile robot. *IEEE J. Robot. Autom.* **1986**, *2*, 14–23. [CrossRef]
13. Brooks, R.; Maes, P.; Mataric, M.; More, G. Lunar base construction robots. In Proceedings of the EEE International Workshop on Intelligent Robots and Systems, Towards a New Frontier of Applications, Ibaraki, Japan, 3–6 July 1990; Volume 1, pp. 389–392. [CrossRef]
14. Sheth, B.D.; Maes, P. Evolving agents for personalized information filtering. In Proceedings of the 9th IEEE Conference on Artificial Intelligence for Applications, Orlando, FL, USA, 1–5 March 1993; pp. 345–352.
15. Maes, P. Modeling Adaptive Autonomous Agents. *Artif. Life* **1993**, *1*, 135–162. [CrossRef]
16. Maes, P.; Darrell, T.; Blumberg, B.; Pentland, A. The ALIVE system: Full-body interaction with autonomous agents. In Proceedings of the Proceedings Computer Animation'95, Geneva, Switzerland, 19–21 April 1995; pp. 11–18. [CrossRef]
17. Benda, M.; Jagannathan, V.; Dodhiawala, R. *On Optimal Cooperation of Knowledge Sources—An Empirical Investigation*; Technical Report BCS-G2010-28; Boeing Advanced Technology Center, Boeing Computing Services: Seattle, WA, USA, 1986.
18. Bratman, M. *Intention, Plans, and Practical Reason*; Harvard University Press: Cambridge, MA, USA, 1987.
19. Georgeff, M.P.; Lansky, A.L. Reactive Reasoning and Planning. In Proceedings of the Sixth National Conference on Artificial Intelligence AAAI, Seattle, WA, USA, 13–17 July 1987.
20. Rao, A.S.; Georgeff, M.P. BDI Agents: From Theory to Practice. In Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95), San Francisco, CA, USA, 12–14 June 1995; pp. 312–319.
21. Fum, D.; Guida, G.; Tasso, C. A Distributed Multi-Agent Architecture for Natural Language Processing. In Proceedings of the 12th Conference on Computational Linguistics, COLING '88, Budapest, Hungary, 22–27 August 1988; Association for Computational Linguistics: Stroudsburg, PA, USA, 1988; Volume 2, pp. 812–814. [CrossRef]
22. Novick, D. Modeling belief and action in a multi-agent system. In Proceedings of the Proceedings [1990], AI, Simulation and Planning in High Autonomy Systems, Tucson, AZ, USA, 26–27 March 1990; pp. 34–41. [CrossRef]
23. Torrance, M.C.; Viola, P.A. The AGENT0 manual. Technical Report STAN-CS-91-1389; Stanford University: Stanford, CA, USA, 1991.

24. Shoham, Y. Agent-oriented programming. *Artif. Intell.* **1993**, *60*, 51–92. [CrossRef]
25. Jennings, N.R. The ARCHON System and its Applications. In Proceedings of the 2nd International Working Conference on Cooperating Knowledge Based System, Keele, UK, 14–17 June 1994.
26. Finin, T.; Fritzson, R.; McKay, D.; Mcentire, R. *KQML as an Agent Communication Language*; Association for Computing Machinery: New York, NY, USA, 1994; pp. 456–463.
27. Austin, J.L. *How to Do Things with Words*; Oxford University Press: New York, NY, USA, 1962.
28. Foundation for Intelligent Physical Agents, Agent Communication Language. 1997. Available online: https://web.archive.org/web/20220308163048/http://www.fipa.org/specs/fipa00018/OC00018.pdf (accessed on 12 April 2023).
29. D'Inverno, M.; Luck, M. Engineering AgentSpeak(L): A formal computational model. *J. Log. Comput.* **1998**, *8*, 233–260. [CrossRef]
30. Jennings, N.; Wooldridge, M. Software Agents. *IEE Review* **1996**, *42*, 17–20. .:19960101. [CrossRef]
31. Wooldridge, M. *What Agents Aren't: A Discussion Paper*; IET: London, UK, 1996.
32. Wooldridge, M. Agent-based software engineering. *IEE Proc. Softw. Eng.* **1997**, *144*, 26–37. [CrossRef]
33. Ferguson, I. Touring Machines: Autonomous agents with attitudes. *Computer* **1992**, *25*, 51–55. [CrossRef]
34. Müller, J.; Pischel, M. *The Agent Architecture InteRRaP: Concept and Application*; Deutsches Forschungszentrum für Künstliche Intelligenz: Kaiserslautern, Germany, 1993.
35. Pomerleau, D.A. ALVINN: An Autonomous Land Vehicle in a Neural Network. In *Proceedings of the Advances in Neural Information Processing Systems*; Touretzky, D., Ed.; Morgan-Kaufmann: San Mateo, CA, USA, 1988; Volume 1.
36. Jordan, R.E.; Andreas, E.L.; Makshtas, A.P. Heat budget of snow-covered sea ice at North Pole 4. *J. Geophys. Res. Ocean.* **1999**, *104*, 7785–7806. [CrossRef]
37. Thrun, S.; Montemerlo, M.; Dahlkamp, H.; Stavens, D.; Aron, A.; Diebel, J.; Fong, P.; Gale, J.; Halpenny, M.; Hoffmann, G.; et al. Stanley: The Robot That Won the DARPA Grand Challenge. In *The 2005 DARPA Grand Challenge: The Great Robot Race*; Buehler, M., Iagnemma, K., Singh, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 1–43. [CrossRef]
38. Epstein, J.; Axtell, R. *Growing Artificial Societies: Social Science from the Bottom Up*; A Bradford Book; MIT Press: Cambridge, MA, USA, 1996.
39. Hammond, R.; Dynamics, E.; Institution, B.; University, J.H. *Endogenous Transition Dynamics in Corruption: An Agent-Based Computer Model*; Number 19 in Working paper (Center on Social and Economic Dynamics); Center on Social and Economic Dynamics: Washington DC, USA, 2000.
40. Axtell, R.; Epstein, J.; Dean, J.; Gumerman, G.; Swedlund, A.; Harburger, J.; Chakravarty, S.; Hammond, R.; Parker, J.; Parker, M. Population Growth and Collapse in a Multiagent Model of the Kayenta Anasazi in Long House Valley. *Proc. Natl. Acad. Sci. USA* **2002**, *99* (Suppl. S3), 7275–7279. [CrossRef] [PubMed]
41. Epstein, J.M. Modeling civil violence: An agent-based computational approach. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 7243–7250. [CrossRef]
42. AgentLink Phase I, 1998. Available online: https://web.archive.org/web/20230610060918/https://cordis.europa.eu/project/id/27225 (accessed on 12 April 2023).
43. AgentLink Phase II, 2000. Available online: https://web.archive.org/web/20230610060927/https://cordis.europa.eu/project/id/IST-1999-29003 (accessed on 12 April 2023).
44. AgentLink Phase III, 2004. Available online: https://web.archive.org/web/20230610061017/https://cordis.europa.eu/project/id/002006 (accessed on 12 April 2023).
45. Sasha, O. *Agreement Technologies*; Law, Governance and Technology Series; Springer: Dordrecht, The Netherlands, 2012. [CrossRef]
46. Los, J.; Schulte, F.; Spaan, M.T.J.; Negenborn, R.R. An Auction-Based Multi-Agent System for the Pickup and Delivery Problem with Autonomous Vehicles and Alternative Locations. In *Proceedings of the Dynamics in Logistics*; Freitag, M., Kinra, A., Kotzab, H., Megow, N., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 244–260.
47. Chen, C.; Koll, C.; Wang, H.; Lindell, M.K. An interdisciplinary agent-based evacuation model: Integrating the natural environment, built environment, and social system for community preparedness and resilience. *Nat. Hazards Earth Syst. Sci.* **2023**, *23*, 733–749. [CrossRef]
48. Clemen, T.; Ahmady-Moghaddam, N.; Lenfers, U.A.; Ocker, F.; Osterholz, D.; Ströbele, J.; Glake, D. Multi-Agent Systems and Digital Twins for Smarter Cities. In Proceedings of the 2021 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, SIGSIM-PADS '21, Virtual Event, 31 May–2 June 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 45–55. [CrossRef]
49. Kozhevnikov, S.; Svitek, M.; Skobelev, P. Smart Grid System for Real-Time Adaptive Utility Management in Smart Cities. In Proceedings of the 13th International Multi-Conference on Complexity, Informatics and Cybernetics (IMCIC 2022), Online, 8–11 March 2022; Volume 1, pp. 4–9 . [CrossRef]
50. Ghribi, C.; Cali, E.; Hirsch, C.; Jahnel, B. Agent-Based Simulations for Coverage Extensions in 5G Networks and Beyond. In Proceedings of the 25th Conference on Innovation in Clouds, Internet and Networks, ICIN 2022, Paris, France, 7–10 March 2022; Zhani, M., Limam, N., Borylo, P., Boubendir, A., dos Santos, C., Eds.; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2022; pp. 1–8. [CrossRef]
51. Xu, J.; Dziong, Z.; Luxin, Y.; Huang, Z.; Xu, P.; Cabani, A. Intelligent multi-agent based C-RAN architecture for 5G radio resource management. *Comput. Netw.* **2020**, *180*, 107418. [CrossRef]

52. Fazio, M.; Pluchino, A.; Inturri, G.; Le Pira, M.; Giuffrida, N.; Ignaccolo, M. Exploring the impact of mobility restrictions on the COVID-19 spreading through an agent-based approach. *J. Transp. Health* **2022**, *25*, 101373. [CrossRef]
53. Bădică, A.; Bădică, C.; Ganzha, M.; Ivanović, M.; Paprzycki, M. Multi-agent Spatial SIR-Based Modeling and Simulation of Infection Spread Management. In Proceedings of the Computational Science—ICCS 2021, Krakow, Poland, 16–18 June 2021; Paszynski, M., Kranzlmüller, D., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 440–453.
54. Adenaw, L.; Lienkamp, M. Multi-Criteria, Co-Evolutionary Charging Behavior: An Agent-Based Simulation of Urban Electromobility. *World Electr. Veh. J.* **2021**, *12*, 18. [CrossRef]
55. Lemiec, M.; Malinowski, K.; Szymoński, M.; Ganzha, M.; Paprzycki, M. Agent-based modelling of car platooning for traffic optimization. In Proceedings of the 2021 4th International Symposium on Agents, Multi-Agent Systems and Robotics (ISAMSR), Batu Pahat, Malaysia, 6–8 September 2021; pp. 130–137. [CrossRef]
56. Pniewski, R.; Sellin, D.; Stankevich, K.; Ganzha, M.; Paprzycki, M. Applying Software Agents to Make City Traffic Management Smarter. In *Proceedings of the Second International Conference on Information Management and Machine Intelligence*; Goyal, D., Gupta, A.K., Piuri, V., Ganzha, M., Paprzycki, M., Eds.; Springer: Singapore, 2021; pp. 651–659.
57. Pniewski, R.; Stankevich, K.; Ganzha, M.; Paprzycki, M. Modelling and Optimizing City Traffic Using an Agent Platform. In *Proceedings of the 2nd International Conference on Artificial Intelligence: Advances and Applications*; Mathur, G., Bundele, M., Lalwani, M., Paprzycki, M., Eds.; Springer Nature: Singapore, 2022; pp. 861–868.
58. Anwar, M.B.; Stephen, G.; Dalvi, S.; Frew, B.; Ericson, S.; Brown, M.; O'Malley, M. Modeling investment decisions from heterogeneous firms under imperfect information and risk in wholesale electricity markets. *Appl. Energy* **2022**, *306*, 117908. [CrossRef]
59. Su, J.; Huang, J.; Adams, S.; Chang, Q.; Beling, P.A. Deep multi-agent reinforcement learning for multi-level preventive maintenance in manufacturing systems. *Expert Syst. Appl.* **2022**, *192*, 116323. [CrossRef]
60. Jabber, A.; Obied, A. Implementing the EBDI model in an E-health system. *Int. J. Nonlinear Anal. Appl.* **2022**, *13*, 1827–1839. [CrossRef]
61. Kim, B.; Lim, C.G.; Lee, S.H.; Jung, Y.J. A Study on the Population Distribution Prediction in Large City using Agent-Based Simulation. In Proceedings of the 2021 23rd International Conference on Advanced Communication Technology (ICACT), PyeongChang, Republic of Korea, 7–10 February 2021; pp. 68–71. [CrossRef]
62. Paré, D.; Shanahan, M.C.; Sengupta, P. *Queering Complexity Using Multi-Agent Simulations*; International Society of the Learning Sciences (ISLS): Boulder, CO, USA, 2020.
63. Hossam, S.; Abd elkader, H.; Khedr, A.; Salem, R. Developing Multiagent E-Learning System-Based Machine Learning and Feature Selection Techniques. *Comput. Intell. Neurosci.* **2022**, *2022*, 2941840 . [CrossRef]
64. Omidshafiei, S.; Kim, D.K.; Liu, M.; Tesauro, G.; Riemer, M.; Amato, C.; Campbell, M.; How, J.P. Learning to Teach in Cooperative Multiagent Reinforcement Learning. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; AAAI'19/IAAI'19/EAAI'19; AAAI Press: Menlo Park, CA, USA, 2019. [CrossRef]
65. Wang, J.; Lv, W.; Jiang, Y.; Qin, S.; Li, J. A multi-agent based cellular automata model for intersection traffic control simulation. *Phys. A Stat. Mech. Appl.* **2021**, *584*, 126356. [CrossRef]
66. Qian, Y.; Barthelemy, J.; Perez, P. Towards Agent-Based Traffic Simulation Using Live Data from Sensors for Smart Cities. In *Proceedings of the Multi-Agent-Based Simulation XXI*; Swarup, S., Barthelemy, B.T.R., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 28–40.
67. Delhoum, Y.; Belaroussi, R.; Dupin, F.; Zargayouna, M. Multi-Agent Activity-Based Simulation of a Future Neighborhood. In *Agents and Multi-Agent Systems: Technologies and Applications 2021: Proceedings of 15th KES International Conference, KES-AMSTA 2021, June 2021*; Springer: Singapore, 2021; pp. 501–510. [CrossRef]
68. Kleinmeier, B.; Köster, G.; Drury, J. Agent-based simulation of collective cooperation: From experiment to model. *J. R. Soc. Interface* **2020**, *17*. [CrossRef]
69. Muravev, D.; Hu, H.; Rakhmangulov, A.; Mishkurov, P. Multi-agent optimization of the intermodal terminal main parameters by using AnyLogic simulation platform: Case study on the Ningbo-Zhoushan Port. *Int. J. Inf. Manag.* **2021**, *57*, 102133. [CrossRef]
70. Panda, M.; Das, B. Multi-agent System of Autonomous Underwater Vehicles in Octagon Formation. In *Intelligent Systems: Proceedings of ICMIB 2020*; Springer: Singapore, 2021; pp. 125–138. [CrossRef]
71. Shah, S.H.H.; Steinnes, O.M.; Gribbestad Gustafsson, E.; Hameed, I. Multi-Agent Robot System to Monitor and Enforce Physical Distancing Constraints in Large Areas to Combat COVID-19 and Future Pandemics. *Appl. Sci.* **2021**, *11*, 7200. [CrossRef]
72. Sepulveda, R.; Alanis, A.; Alarcón, M.A.; Velazquez, D.; Baltazar, R. Intelligent Agent for Actuator Control in a Robot (IA-ACR). In Proceedings of the KES-AMSTA, Virtual Conference, 14–15 June 2021.
73. Bourceret, A.; Amblard, L.; Mathias, J.D. Governance in social-ecological agent-based models: A review. *Ecol. Soc.* **2021**, *26*, 38. [CrossRef]
74. Jager, W. Using agent-based modelling to explore behavioural dynamics affecting our climate. *Curr. Opin. Psychol.* **2021**, *42*, 133–139.

75. Reguly, I.; Csercsik, D.; Juhasz, J.; Tornai, K.; Bujtar, Z.; Horvath, G.; Keomley-Horvath, B.; Kos, T.; Cserey, G.; Ivan, K.; et al. Microsimulation based quantitative analysis of COVID-19 management strategies. *PLoS Comput. Biol.* **2022**, 18, e1009693. [CrossRef] [PubMed]
76. Lorig, F.; Johansson, E.; Davidsson, P. Agent-Based Social Simulation of the Covid-19 Pandemic: A Systematic Review. *J. Artif. Soc. Soc. Simul.* **2021**, *24*, 1–5. [CrossRef]
77. Sato, K.; Sugawara, T. Multi-Agent Task Allocation Based on Reciprocal Trust in Distributed Environments. In *Proceedings of the Agents and Multi-Agent Systems: Technologies and Applications 2021*; Jezic, G., Chen-Burger, J., Kusek, M., Sperka, R., Howlett, R.J., Jain, L.C., Eds.; Springer: Singapore, 2021; pp. 477–488.
78. Noorunnisa, S.; Jarvis, D.; Jarvis, J.; Rönnquist, R. A Conceptual Model for Human-Agent Teams. In *Agents and Multi-Agent Systems: Technologies and Applications 2021: Proceedings of 15th KES International Conference, KES-AMSTA 2021, June 2021*; Springer: Singapore, 2021; pp. 17–26. [CrossRef]
79. Gracia-Lázaro, C.; Dercole, F.; Moreno, Y. Dynamics of economic unions: An agent-based model to investigate the economic and social drivers of withdrawals. *Chaos Solitons Fractals* **2022**, *160*, 112223. [CrossRef]
80. Ha, T.; Lee, S. Examination of Bitcoin Exchange Through Agent-Based Modeling: Focusing on the Perceived Fundamental of Bitcoin. *IEEE Trans. Eng. Manag.* **2022**, *69*, 1294–1307. [CrossRef]
81. Fraunholz, C.; Kraft, E.; Keles, D.; Fichtner, W. Advanced price forecasting in agent-based electricity market simulation. *Appl. Energy* **2021**, *290*, 116688. [CrossRef]
82. Mutlag, A.A.; Ghani, M.K.A.; Mohammed, M.A.; Lakhan, A.; Mohd, O.; Abdulkareem, K.H.; Garcia-Zapirain, B. Multi-Agent Systems in Fog–Cloud Computing for Critical Healthcare Task Management Model (CHTM) Used for ECG Monitoring. *Sensors* **2021**, *21*, 6923. [CrossRef] [PubMed]
83. Akbari, Z.; Unland, R. *A Holonic Multi-Agent System for the Support of the Differential Diagnosis Process in Medicine*; Universität Duisburg-Essen: Duisburg, Germany, 2021.
84. Parv, L.; Deaky, B.; Marius Daniel, N.; Oancea, G. Agent-Based Simulation of Value Flow in an Industrial Production Process. *Processes* **2019**, *7*, 82. [CrossRef]
85. Wan, G.; Dong, X.; Dong, Q.; He, Y.; Zeng, P. Design and implementation of agent-based robotic system for agile manufacturing: A case study of ARIAC 2021. *Robot. Comput.-Integr. Manuf.* **2022**, *77*, 102349. [CrossRef]
86. Lee, J.; Shin, S.; Park, M.; Kim, C. Agent-Based Simulation and Its Application to Analyze Combat Effectiveness in Network-Centric Warfare Considering Communication Failure Environments. *Math. Probl. Eng.* **2018**, *2018*, 2730671. [CrossRef]
87. Parayil, A.; George, J. Distributed Tracking and Circumnavigation Using Bearing Measurements. In Proceedings of the ICASSP 2020—2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 4885–4889. [CrossRef]
88. George, J.; Yilmaz, C.T.; Parayil, A.; Chakrabortty, A. A Model-Free Approach to Distributed Transmit Beamforming. In Proceedings of the ICASSP 2020—2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 5170–5174. [CrossRef]
89. Massive Software Applications and Products. Available online: https://web.archive.org/web/20230607161204/https://www.massivesoftware.com/ (accessed on 12 April 2023).
90. Unity Technologies Machine Learning. Available online: https://web.archive.org/web/20230610060410/https://unity.com/products/machine-learning-agents (accessed on 12 April 2023).
91. Booth, J.; Booth, J. Marathon Environments: Multi-Agent Continuous Control Benchmarks in a Modern Video Game Engine. *arXiv* **2019**, arXiv:1902.09097. https://doi.org/10.48550/arXiv.1902.09097.
92. DeepMotion Motion Brain. Available online: https://web.archive.org/web/20230317170556/https://deepmotion.com/ai-motion-brain (accessed on 12 April 2023).
93. Safia, R.; Hachicha, H.; Zagrouba, E. Multi-Agent-Based Framework for Resource Allocation in Cloud Computing. In *Agents and Multi-Agent Systems: Technologies and Applications 2021: Proceedings of 15th KES International Conference, KES-AMSTA 2021, June 2021*; Springer: Singapore, 2021; pp. 427–437. [CrossRef]
94. Deochake, S.; Mukhopadhyay, D. An Agent-Based Cloud Service Negotiation in Hybrid Cloud Computing. In *Proceedings of the ICT Systems and Sustainability*; Tuba, M., Akashe, S., Joshi, A., Eds.; Springer: Singapore, 2021; pp. 563–572.
95. Zhang, Y.; Wen, G.; Rahmani, A.; Peng, Z.; Hu, W. Cluster consensus of multi-agent systems with general linear and nonlinear dynamics via intermittent adaptive pinning control. *Trans. Inst. Meas. Control* **2021**, *43*, 1337–1346. [CrossRef]
96. Bettini, L.; Bourr, K.; Pugliese, R.; Tiezzi, F. Programming Multi-robot Systems with X-KLAIM. In *Proceedings of the Leveraging Applications of Formal Methods, Verification and Validation. Adaptation and Learning*; Margaria, T., Steffen, B., Eds.; Springer Nature Switzerland: Cham, Switzerland, 2022; pp. 283–300.
97. Mariani, S.; Omicini, A. TuCSoN on Cloud: An Event-Driven Architecture for Embodied / Disembodied Coordination. In *Proceedings of the Algorithms and Architectures for Parallel Processing*; Aversa, R., Kołodziej, J., Zhang, J., Amato, F., Fortino, G., Eds.; Springer International Publishing: Cham, Switzerland, 2013; pp. 285–294.
98. Mamei, M.; Zambonelli, F. Programming pervasive and mobile computing applications: The TOTA approach. *ACM Trans. Softw. Eng. Methodol.* **2009**, *18*, 15:1–15:56. [CrossRef]
99. Castelli, G.; Mamei, M.; Rosi, A.; Zambonelli, F. Engineering Pervasive Service Ecosystems: The SAPERE Approach. *ACM Trans. Auton. Adapt. Syst.* **2015**, *10*, 1:1–1:27. [CrossRef]

100. Serenko, A.; Detlor, B. *Agent Toolkits: A General Overview of the Market and an Assessment of Instructor Satisfaction with Utilizing Toolkits in the Classroom*; DeGroote School of Business: Hamilton, ON, Canada, 2002.
101. Nguyen, J.; Powers, S.T.; Urquhart, N.; Farrenkopf, T.; Guckert, M. An overview of agent-based traffic simulators. *Transp. Res. Interdiscip. Perspect.* **2021**, *12*, 100486. [CrossRef]
102. Huang, J.; Cui, Y.; Zhang, L.; Tong, W.; Shi, Y.; Liu, Z. An Overview of Agent-Based Models for Transport Simulation and Analysis. *J. Adv. Transp.* **2022**, *2022*, 1252534. [CrossRef]
103. Negahban, A.; Yilmaz, L. Agent-based simulation applications in marketing research: An integrated review. *J. Simul.* **2014**, *8*, 129–142. [CrossRef]
104. Mualla, Y.; Bai, W.; Galland, S.; Nicolle, C. Comparison of Agent-based Simulation Frameworks for Unmanned Aerial Transportation Applications. *Procedia Comput. Sci.* **2018**, *130*, 791–796.
105. Niazi, M.; Hussain, A. Agent-Based Tools for Modeling and Simulation of Self-Organization in Peer-to-Peer, Ad Hoc, and Other Complex Networks. *Commun. Mag. IEEE* **2009**, *47*, 166 – 173. [CrossRef]
106. Groeneveld, J.; Müller, B.; Buchmann, C.; Dressler, G.; Guo, C.; Hase, N.; Hoffmann, F.; John, F.; Klassert, C.; Lauf, T.; et al. Theoretical foundations of human decision-making in agent-based land use models—A review. *Environ. Model. Softw.* **2017**, *87*, 39–48. [CrossRef]
107. Ricordel, P.m.; Demazeau, Y. From Analysis to Deployment: A Multi-Agent Platform Survey. In *Engineering Societies in the Agents World: Proceedings of the First International Workshop, ESAW 2000 Berlin, Germany, 21 August 2000*; Springer: Berlin/Heidelberg, Germany, 2000. [CrossRef]
108. Antelmi, A.; Cordasco, G.; D'Ambrosio, G.; De Vinco, D.; Spagnuolo, C. Experimenting with Agent-Based Model Simulation Tools. *Appl. Sci.* **2023**, *13*, 13. [CrossRef]
109. Dorri, A.; Kanhere, S.S.; Jurdak, R. Multi-Agent Systems: A Survey. *IEEE Access* **2018**, *6*, 28573–28593. [CrossRef]
110. Railsback, S.F.; Lytinen, S.L.; Jackson, S.K. Agent-based Simulation Platforms: Review and Development Recommendations. *Simulation* **2006**, *82*, 609–623. [CrossRef]
111. Bordini, R.; Lars, B.; Dastani, M.; Seghrouchni, A.; Gómez-Sanz, J.; Leite, J.; O'Hare, G.; Alexander, P.; Ricci, A. A Survey of Programming Languages and Platforms for Multi-Agent Systems. *Informatica* **2006**, *30*, 33–44.
112. Allan, R. *Survey of Agent Based Modelling and Simulation Tools*; Science & Technology Facilities Council: New York, NY, USA, 2009; p. 1362-0207.
113. Abar, S.; Theodoropoulos, G.K.; Lemarinier, P.; M.P.O'Hare, G. Agent Based Modelling and Simulation tools: A review of the state-of-art software. *Comput. Sci. Rev.* **2017**, *24*, 13–33. [CrossRef]
114. Cardoso, R.; Ferrando, A. A Review of Agent-Based Programming for Multi-Agent Systems. *Computers* **2021**, *10*, 16. [CrossRef]
115. Lugrin, B.; Pelachaud, C.; Traum, D., Eds. *The Handbook on Socially Interactive Agents: 20 Years of Research on Embodied Conversational Agents, Intelligent Virtual Agents, and Social Robotics Volume 2: Interactivity, Platforms, Application*, 1st ed.; Association for Computing Machinery: New York, NY, USA, 2022; Volume 48.
116. Helle, P.; Feo-Arenis, S.; Strobel, C.; Shortt, K. Agent-Based Modelling and Simulation of Decision-Making in Flying Ad-Hoc Networks. In *Proceedings of the Advances in Practical Applications of Agents, Multi-Agent Systems, and Complex Systems Simulation. The PAAMS Collection*; Dignum, F., Mathieu, P., Corchado, J.M., De La Prieta, F., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 242–253.
117. Nunes, I. Capability Relationships in BDI Agents. In Proceedings of the The 2nd International Workshop on Engineering Multi-Agent Systems (EMAS 2014) at AAMAS 2014, Paris, France, 5–6 May 2014; pp. 56–72.
118. Chen, F.; Zhao, Q.; Cao, M.; Chen, J.; Fu, G. Adaptive Agent-Based Modeling Framework for Collective Decision-Making in Crowd Building Evacuation. *J. Shanghai Jiaotong Univ. (Sci.)* **2021**, *26*, 522–533. [CrossRef]
119. Le, N.T.T. Multi-agent reinforcement learning for traffic congestion on one-way multi-lane highways. *J. Inf. Telecommun.* **2023**, 1–15. [CrossRef]
120. Bădică, A.; Bădică, C.; Buligiu, I.; Ciora, L.I. DEVS Modeling and Simulation Using BDI Agents: Preliminary Considerations. In Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics, Novi Sad, Serbia, 25–27 June 2018; pp. 1–8. [CrossRef]
121. Murić, G.; Tregubov, A.; Blythe, J.; Abeliuk, A.; Choudhary, D.; Lerman, K.; Ferrara, E. Large-Scale Agent-Based Simulations of Online Social Networks. *Auton. Agents Multi-Agent Syst.* **2022**, *36*, 38. [CrossRef]
122. Freeman, C.D.; Frey, E.; Raichuk, A.; Girgin, S.; Mordatch, I.; Bachem, O. Brax—A Differentiable Physics Engine for Large Scale Rigid Body Simulation. *arXiv* **2021**, arXiv:2106.13281.
123. Areekkara, S.; Kumar, R.; Bansal, R.C. An Intelligent Multi Agent based Approach for Autonomous Energy Management in a Microgrid. *Electr. Power Compon. Syst.* **2021**, *49*, 18–31. [CrossRef]
124. Pianini, D.; Montagna, S.; Viroli, M. Chemical-oriented simulation of computational systems with ALCHEMIST. *J. Simul.* **2013**, *7*, 202–215. [CrossRef]