International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252

A Review of Agent-Based Programming for Multi-Agent Systems

Dr. M. Purushotham

Date of Submission: 10-08-2023 Date of Acceptance: 20-08-2023

ABSTRACT:
Intelligent and independent agents is a subarea of emblematic artificial intelligence where these agents decide, either reactively or proactively, upon a course of action by logic about the information that's available about the world(including the terrain, the agent itself, and other agents). It encompasses a multitude of ways, similar as concession protocols, agent simulation,multi-agent confabulation,multi-agent planning, and numerous others. In this paper, we concentrate on agent programming and we give a methodical review of the literature in agent- grounded programming formulti-agent systems. In particular, we bandy both stager(still maintained) and new agent programming languages, their extensions, work on comparing some of these languages, and operations set up in the literature that make use of agent programming

Keywords: agent-based programming; multi-agent systems; agent programming languages

I. INTRODUCTION

Multi-Agent Systems(MASs)(1) are a well established branch of Artificial Intelligence AI). Indeed though they're fairly youthful with respect to further archetypal exploration areas, MASs have a rich history; in 1995(2) agent technology was recognised as a fleetly developing exploration area and one of the fastest growing areas of information technology. Such a statement is still true currently, where one can find plenitude of exploration papers, tools, and conferences whose end is to advance the exploration in the area. Despite this, MASs are not as extensively used as they could be. Considering the agent programming aspect of MASs, according to(3), the crucial reason is that there's little incitement for inventors to switch to current Agent Programming Languages(APLs), as the behaviours that can be fluently programmed are sufficiently simple to be implementable in mainstream languages with only a small outflow in rendering time. This, amongst the presence of too numerousunorganised options available, doesn't help agent- grounded programming languages and tools to be picked fromnon-expert druggies.

An intelligent agent(4) can be generalised as a computerised reality that's suitable to reason(rational/ cognitive), to make its own opinions singly(independent), to unite with other agents when necessary(social), to perceive the environment in which it operates and reply to it meetly(reactive), and eventually, to take action in order to achieve its pretensions(visionary). An agent- grounded(or agent- acquainted) system is a system where the agents are the main realities, treated as first- class abstractions. From a programming perspective, the same logic can be followed. In particular, by using a comparison, we can say that agents are to Agent- acquainted Programming(AOP) languages as objects are to Object- acquainted Programming(OOP) languages. In an agent- grounded programming language, agents are the structure blocks, and programs are attained by programming their behaviours(how an agent reasons), their pretensions(what an agent aims to achieve) and their interoperation(how agents unite to break a task).

Agents are well- suited to be used in operations involving distributed or concurrent calculation or when communication is needed between different factors. For this reason, agent technology is useful in operations that reason about dispatches objects entered over a network. By conserving their processing state and the state of the world around them, agents are also immaculately suited to robotization operations. also, independent agents can operate without stoner intervention and can be used in operations similar as factory/ process robotization, workflow operation, robotics, and others. Another advantage of agent- grounded programming is that due to the logic cycle present in agents, it is also possible to give explanations about the opinions that an agent has made.

The end of this paper is to review the rearmost work in agent- grounded programming, to help both experts andnon-experts druggies having a better grasp over the current state of the art in

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252

agent- grounded programming technologies, and to identify unborn directions of exploration in this area. In particular, we concentrate on the rearmost agent programming languages, platforms and fabrics for the development of MASs. Both theoretical and practical papers are taken into account, and we also compactly bandy recent extensions, being comparisons, and operations of agent- base programming.

With respect to other reviews and checks on APLs in literature(3,5 – 10), our review focuses on recent developments and considers workshop presenting new APLs, as well as works fastening on extending or comparing being APLs. also, we consider both theoretical and practical aspects; this helps to have a better understanding of the reality gap between theoretical and practical APLs. Eventually, with respect to former reviews, we concentrate on the entire class of APLs, and not on a specific area of APLs as in(6), where only the engineering aspects are considered, or in(7), where only APLs platforms are considered, or in(8), where only agent- grounded simulation literature is analysed, or in(3,5), where their focus is in a specific model of agency(Belief- Desire- Intention — BDI).

This paper is structured as follows. A brief history on agent- grounded programming is given in Section 2. In Section 3, the methodical review process followed in this paper is presented. Section 4 contains the review findings with the papers set up in our hunt of the literature. In Section 5, the review's results are bandied and unborn directions are suggested. Eventually, in Section 6, the conclusions of the paper are reported.

II. HISTORY ON AGENT-BASED PROGRAMMING

In 1993, Agent- acquainted Programming was first introduced(11) as a specialisation of Object- acquainted Programming. Most specially, it discusses the notion of the internal state of an agent, conforming of its information, opinions, and capabilities. This work also describes agent programs in the AGENT- 0 practitioner(enforced in the Lisp language) and their communication using speech act proposition, the ultimate is still used to define agent communication in several contemporary agent programming languages. Over the times, numerous logic and cognitive models have been developed for agent- grounded programming. In this section, we bandy three particular models that have been abecedarian in the design of numerous agent programming languages in the history and that are still being used in new languages these days Procedural logic System(PRS), BDI, and Situation Calculus.

The Procedural logic System(PRS)(12)(enforced in Lisp) defines a system able of logic about processes, that is, procedural forms of knowledge. An agent in this system is also suitable to use these procedures to elect intentions for achieving particular pretensions. Unlike in conventional programming languages, these procedures aren't invoked a priori, but they're touched off when they're suitable to contribute towards some thing or to reply to some situation. While participating some parallels to AI itineraries of the time, its main difference is that it performs partial hierarchical planning in the sense that it interacts with a dynamic terrain during the logic process, rather of generating a plan for a stationary terrain.

The Belief- Desire- Intention(BDI) model(13,14) consists of a logic process that aids the decision- timber of opting an applicable action towards the achievement of some thing. Its three internal stations are belief — knowledge that the agent believes about its terrain, itself, and other agents; desire — the asked countries that the agent wants to achieve; and intention — a sequence of way towards the achievement of a desire. These internal stations independently represent the information, motivational, and deliberativestates of the agent. The workflow in a general BDI system is shown in Figure 1 and works as such a belief modification function receives input information from the terrain(e.g., detectors), and it's responsible for streamlining the belief base. This update can induce further options that can come current solicitations grounded on the belief base and the intentions base. A sludge is responsible for streamlining the intentions base, taking into account its former state and the current belief base and desire base. Eventually, an intention is chosen to be carried out as an action by the agent. BDI is the most popular model of agency, it has been and continues to be used in numerous agent programming languages. AgentSpeak(L)(15) is a language that serves as an abstraction of enforced BDI systems that can be used to interpret agent programs as cornucopia- clause sense programs. The proposition behind this language has been enforced as a base for numerous APLs.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252

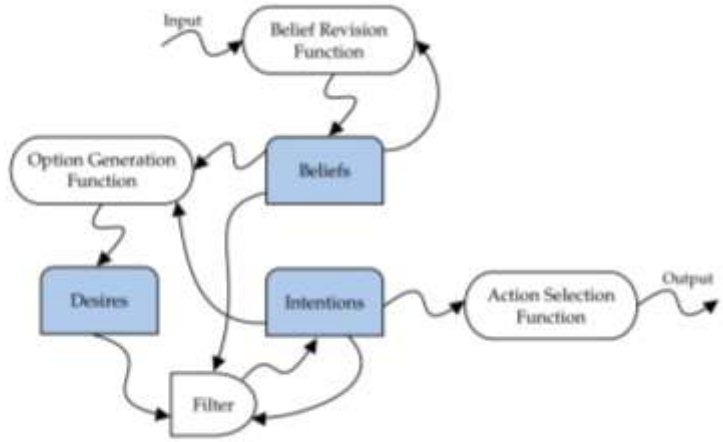


Figure 1. The BDI model.

Situation Calculus(16) is a first order language designed to represent changes in dynamic surroundings. A situation is a first order term representing a sequence of conduct. An original situation is when no conduct have passed yet. The function do(a, s) results in a successor situation to s after executing the action a, analogous to state transition systems. Dynamic surroundings play an important part in agent-grounded programming, and as similar, Situation Calculus has been used to model how the world changes as a result of executing conduct

As we will see in Section 4, there are numerous other models that have inspired agentbased programming languages, still, these three were the most influential in the once history of agent- grounded programming. Some agent languages partake parallels or indeed blend generalities from other programming paradigms, similar as procedural, imperative, objectoriented, functional, actor, concurrent, and so on. Comparing the differences or going into detail about these other paradigms is out of compass of this review, but we still consider agent languages that combine generalities from different paradigms.

Historically, agent- grounded programming has been used in a myriad of practical operations, similar as distributed control of electric power grids(17), governance of room allocation(18) and of automated machine- to-machine operations(e.g., business redirection)(19), and detecting sequestration violations(20). In Section4.4, we cover the more recent attempts in using APLs for programming practical operations.

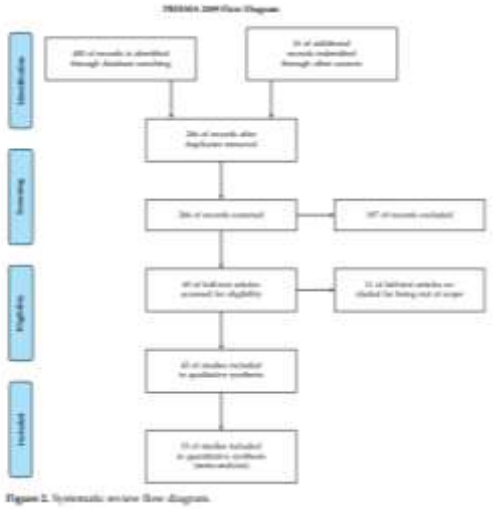
III. REVIEW METHODOLOGY

We performed a methodical review of the literature in agent programming languages over the once 5 times(2015 – 2020). A illustration illustrating our review methodology is shown in Figure 2. For each searched term, we considered the first 10 runners(100 entries) recaptured by Google Scholar(ordered by applicability), for a aggregate of 400 entries(including duplicates).

The terms used in our hunt were

- Agent- Grounded Programming Languages
- Agent- Grounded Programming Extensions
- Agent- Grounded Programming Comparison
- Agent- Grounded Programming operations

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252



After removing duplicates, we had 250 remaining papers. To these, we added 16 entries from external sources; substantially old references that would not appear in the hunt, plus a many others set up in paper citations and other sources. Out of the 16 external entries, eight were influential APLs that have been developed before 2015 and have been streamlined lately(i.e., 2017 – 2020).

IV. REVIEW FINDINGS ON AGENT-BASED PROGRAMMING FOR MAS

In this section, we cover all of the exploration set up in our methodical review of the literature. We start with the agent programming languages and their extensions, also continue to bandy the being comparisons in agent- grounded programming, and close the section with a brief review of operations in the area.

1.1. Agent Programming Languages

We report the agent programming languages set up in the methodical review in Table 1. As we indicated to in Section 2, we can see that BDI is easily the most popular model ofagency, being used in 7 out of the 15 languages. The maturity of the enforced languages have been enforced in Java, most likely to take advantage of thecross-platform of Java through its Java Virtual Machine. The table contains only the high-position general- purpose languages that can be used to develop sphere independent MASs. While other approaches similar as Agent- Grounded Modelling and Simulation(ABMS) and cognitive agents in robotics are not included in the table, we compactly report some of the new exploration that has been done in those areas further below.

Table 1. A collection of recent(or lately streamlined) agent programming languages. Languages that have no intimately available perpetration are represented with ✗. In case there are multiple perpetration branches, the Last streamlined column refers to the last update in the master branch.

APL	Model	Implementation (Language-Link)	Last Updated
ASTRA	BDI	Java https://gitlab.com/astra-language	6 November 2020
Chromar	rule-based	Haskell https://github.com/azardilis/Chromar	14 June 2020
GOAL	rule-based	Java https://goalapl.atlassian.net/wiki/spaces/GOAL/overview	15 December 2020
Gwendolen	BDI	Java https://github.com/mcapl/mcapl	7 December 2020
JaCaMo	BDI, organisation, environment	Java https://github.com/jacamo-lang/jacamo	20 September 2020

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252

JADE	FIPA	Java https://jade.tilab.com/	8 June 2017
JADEL	DSL, interaction	Java/Jade ✗	✗
Jadex	mixed, BDI and OOP	Java https://github.com/actoron/jadex	10 January 2021
Jadescript	DSL, scripting	Java/Jade ✗	✗
Jason	BDI	Java https://github.com/jason-lang/jason	12 November 2020
LightJason	BDI	Java https://github.com/LightJason/	29 December 2020
PLACE	BDI, HTN	✗	✗
PLASA	Wait-Look-Compute-Move	Java ✗	✗
RMAS	database-centric, CPS	Matlab/SQLite ✗	✗
SARL	DSL	Java https://github.com/sarl/sarl	4 January 2021

1.1.1. General-Purpose APLs

A check on agent- acquainted programming from the software engineering perspective can be set up in(6). One of the main challenges reported in the check for APL inventors is the need to bridge the cognitive gap that exists between the generalities bolstering mainstream languages and those bolstering AOP. In(21) the authors try to fill this gap fastening on understanding the relationship between AgentSpeak(L)(15) and OOP with the thing of trying to reduce the perceived cognitive gap. Such a work proposes a new statically compartmented agent programming language entitled ASTRA.

In(22) the authors present Chromar, a rule- grounded memorandum with stochastic semantics yielding a nonstop time Markov chain. Chromar is bedded in Haskell, this givesit an increased suggestive power, and fits with the vacuity of rich types. In Chromar, rules are first-order abstractions that can both describe a(conceivably partial) geste of an individual agent and a synchronised action of two or further agents.

Thing(23) is a declarative agent programming language that uses knowledge base beliefs and pretensions to support the decision-timber of its cognitive agents. Despite participating analogous generalities with the BDI model(beliefs, solicitations pretensions), the thing language is more centred towards rule- grounded decision- timber. Agents programs are written in GOAL's specific syntax, but the knowledge of the agent(e.g., rules) are generally represented in Prolog.

Jason(24) is an extension of the AgentSpeak(L) language, grounded on the BDI agent model. Agents in Jason reply to events in the system by executing conduct on the terrain, according to the plans available in each agent's plan library. One of the extensions in Jason is the addition of Prolog- suchlike rules that can be added and used in the belief base of agents.

The JaCaMoplatform(25,26) is composed of three technologies, Jason, CArtAgO(27), and Moise(28), each representing a different abstraction position. Jason is used for programming the agent position, CArtAgO is responsible for the terrain position, and Moise for the organisation position. JaCaMo integrates these three technologies by defining a semantic link among generalities in different situations of abstraction(agent, terrain, and organisation). The end result is the JaCaMo MAS development platform. It provides high- position first- class support for developing agents, surroundings, and organisations, allowing the development of more complexmulti-agent systems.

Gwendolen(29) originally started as a small subset of Jason in the expedients of developing empirical agent programs, but has since grown into its own syntax and semantic. Because it is a language that has been erected to support agent verification from the ground up, it is limited in what features it can support, still, the basics of AgentSpeak(L) and BDI are all present. There's a vast literature in verification of agent programs and Mamas, but we consider them out of compass for this review. Gwendolen, piecemeal

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252

from being empirical , is still a feasible language for developing general- purposeMASs.

Wanton(30) is an open source platform for the development of peer- to- peer agent grounded operations. Besides the agent abstraction, it also provides task prosecution and composition model, peer- to- peer agent communication grounded on asynchronous communication end, and a unheroic runner service that supports the publish and subscribe discovery medium. JADEbased systems can be distributed across machines with different functional systems, and has been used by numerous languages(e.g., Jason and JaCaMo) as a distribution structure.

In(31) the authors present JADEL(JADE Language), an extension of JADE that provides support for the construction of agents and Mamas on top of JADE without having to use Java directly; latterly, in(32), the authors present Jadescript, an extension of JADEL. Jadescript is characterised by a strong suggestive syntax largely inspired by ultramodern scripting languages in order to promote readability and to make agent programs more analogous to pseudocode.

Jadex(33) allows the programming of intelligent software agents in XML and Java. The agent abstraction is grounded on the BDI model, and provides several features similar as a runtime structure for agents, multiple commerce styles, simulation support, automatic overlay network conformation, and an expansive runtime tool suite.

LightJason, a largely scalable Java-grounded platform for BDI agent- acquainted programming and simulation is presented in(34). LightJason is grounded on a sense language which extends AgentSpeak(L) with lambda-expressions,multi-plan and- rule description, unequivocal form conduct,multi-variable assignments, resemblant prosecution, and thread-safe variables. Indeed however the language is inspired by AgentSpeak(L) and Jason, LightJason is enforced from scrape.

In(35) an AOP language called Planning grounded Language for Agents and Computational surroundings(PLACE) adds AI planning capability to agents. PLACE has a syntactic structure close to BDI, while the planning is done in a Hierarchical Task Network(HTN) diary. In discrepancy to other AOP languages, conduct in PLACE have durations associated to them, therefore, taking the diary to be suitable to handle temporal information. Agents in PLACE have the capability to recover from failures by conforming their conditioning to the new situations. For this purpose, a plan repairing medium is added that repairs a plan if the unexpected events in the terrain beget the plan to come unfeasible

In(36), the Programming Language for Synchronous Agents(PLASA) is proposed. PLASA is platform-independent and facilitates a rapid-fire perpetration of united operations on multiple physical robots and in dynamic surroundings. Basically, PLASA tools a variant of the stay- Look- cipher- Move model proposed in(37), where robots move synchronously. It's designed as a high- position programming language, which allows druggies to specify the instructions to be performed by robots in a mortal-readable language.

Relational ModelMulti-Agent System(RMAS)(38) is a database- centric approach formulti-agent systems suitable for the personification of logic and control in CyberPhysical Systems(CPS). original perpetration of RMAS is proposed by the coupling of the Matlab terrain and the SQLite database language.

SARL's(39) focus is to give an extensible language that's equipped with the minimal quantum of generalities(i.e., crucial generalities) needed to support AOP. The language aims to give abstractions for concurrency, distribution, commerce, decentralisation, reactivity, autonomy, and dynamic reconfiguration. To do so it isn't grounded in any model, but rather it creates its own sphere-Specific Language(DSL) in order to give a reduced and further featherlight core.

1.1.2. ABMS, Robotics, and Others

As recognisedin(40), there's a gap between Agent- acquainted Software Engineering (AOSE) methodologies and the development of ABMS. To overcome this issue, in(41) an AOSE process called Process for Developing Effective Agent- Grounded Simulators(PEABS) is proposed. It uses the INGENIAS methodology(42) for modelling the specification and designing its structure. It applies an adaption frame that allows ABMS inventors to gain simulations with a high effectiveness for large quantities of data. Another approach for developing ABMS is presented in(43,44), where the authors propose a new cognitive agent armature grounded on the BDI model and integrated into the GAMA modelling language(45). With respect to former integration works between BDI and ABMS, in(43) the armature proposed aims to be flexible and easy to use fornon-expert druggies. Another work which aims to integrate BDI and ABMS is presented in(46),

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252

where the authors present a frame that allows BDI cognitive agents to be bedded in an ABMS system. Compared to(43), reference(46) is more general since its ideal is to integrate any BDI- grounded system with ABMS. The only demand is that the percepts(or environmental compliances/ events) of interest to each agent and the conduct that the agent may execute in the simulation terrain can be linked a priori.

ALLEGRO(= ALGOL in PREGO(47,48)) is a programming formalism grounded on belief armature for stochastic disciplines which is intended as an volition to GOLOG(49) for high-position control in robotic operations. Another language which is grounded in GOLOG and the situation math is introduced in(50), where a prototype perpetration of Yet Another GOLOG practitioner(YAGI), an action- grounded robot and agent programming language, is presented. YAGI offers tapes for popular robotics fabrics similar as Robot Operating System(ROS)(51) and Fawkes.

In(52) the authors present a Cognitive Affective Agent Programming Framework CAAF), a frame grounded on the belief- desire proposition of feelings that enables the calculation of feelings for cognitive agents(i.e., making them cognitive affective agents). The authors present semantics showing the programming constructs of these agents. With these constructs, a programmer can make an agent program with cognitive agents that automatically cipher feelings during runs.

4.2. Agent Programming Languages Extensions

In the former section, we reported workshop presenting new APLs(2015 – 2020), along with workshop presenting the most influential APLs(≤ 2015) that are still maintained. Now, we consider the most influential workshop that have extended being APLs. We relate to APL extensions as workshop that have changed being APLs internally, either by adding new features or by erecting on top of being APLs, for illustration, by customising the APL for a new and specific script

An enhanced interpretationofMulti-Agent System for Competitive Electricity requests MASCEM)(53) is presented in(54). This extended interpretation of the MASCEM simulator points at supporting the integration of new and reciprocal models. The facilitation in accommodating different tools and mechanisms is handed by important structural perpetration opinions, making MASCEM suitable to deal with the constantly changing and largely demanding terrain of electricity requests. In particular, the new extension of MASCEM brings the use of ontologies to support players ' dispatches.

In(55), the authors present TABSAOND, an extension of PEABS(41). The main difference between TABSAOND and PEABS is that the former focuses on the design and perpetration of the decision- making processes in non deterministic scripts. In addition, simulators are now stationed as mobile apps and online tools.

In(56) a conservative synchronisation model is proposed for the SARL language and its runtime platform Janus. Since Janus doesn't make any supposition on the ordering of the events that are changed by the agents, it isn't possible to use the Janus platform for agent- grounded simulation involving time without furnishing the platform with a specific synchronisation medium. A model for such a medium is described in their extension

The authors of(57) propose new programming constructs for integrating an advanced yet rule grounded emotion model, EMIA(58), in line with the 2APL(59) agent language. The combination of both has been carried out by reconsidering the syntax, semantics and deliberation cycle of 2APL. This combination substantially focuses on event- grounded emotion generation, and the performing simulation shows high believability in the feelings expressed by the agent when responding to the real life scripts.

ARGO(60) is a customised Jason armature for programming bedded robotic agents using the Javino middleware and perception pollutants.

In(61), the authors show how procedural reflection in the agent programming language meta- APL(62) can be used to allow a straightforward perpetration of some of the way in the deliberation cycle of a BDI agent, by allowing both agent programs and the agent's deliberation strategy to be decoded in the same programming language.

An extension(63) to Jason and Gwendolen allows the agents in these languages to communicate with ROS, therefore supporting the programming of independent agents that can control and perform high- position decision-making in robotic operations developed in ROS. The extension is done through an interface that's used as the terrain between the agent and ROS, and the communication between the terrain and ROS bumps is performed using the rosbridge library. The main difference between their work and once attempts at extending traditional APLs to support ROS is that their approach requires no fresh variations in either of the two APLs or ROS, making it usable and movable to different

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252

performances of these tools. also, in(64) a frame for using Jason with ROS in bedded systems is presented and a new armature is introduced to support lower- position relations between ROS and the agent.

Eventually, in(65) a model for a BDI agent programming frame integrating underpinning literacy and an perpetration grounded on the Jason programming language are introduced. The approach supports the design of BDI agents where some plans can be explicitly programmed and others rather can be learned by the agent during the development/ engineering stage.

4.3. Agent Programming Languages Comparison

From the exploration described in the former two sections we can observe that there are numerous APLs for developing Mamas available in the agent- grounded programming community. Unfortunately, veritably frequently the evaluation of a language is incompletely or indeed fully missing. Some studies similar as(66) have been done in the history to compare agent languages with other paradigms, in that case the comparison was with actor- grounded languages. In their results the authors have shown that agent languages(specifically Jason in that work) can indeed be competitive with further featherlight languages similar as actors.

In(67), the authors present an evaluation frame for assessing being or recently defined sphere-specific modelling languages for MASs. The evaluation targets both the language and the corresponding tools and provides both qualitative and quantitative results.

A comparison between the pseudocode of a well- known algorithm for working distributed constraint satisfaction problems and the perpetration of such an algorithm in JADEL is shown in(68).

The work in(69) focuses on comparing resemblant platforms that supportmulti-agent simulations and their prosecution on high performance coffers as resemblant clusters.

The authors of(70) perform a methodical evaluation of ABMS approaches secerning the generalities of how complex the model geste is and how complicated the model structure is, and illustrate thenon-linear relationship between them. also, they estimate the trade- offs between simple(frequently theoretical) models and complicated(frequently empirically- predicated) models.

4.4. Agent-Based Applications

In this section we list some of the rearmost operations using agent- grounded programming. Our thingthen's to show the wide variety of operation disciplines that agents can be useful in, therefore, this list isn't total and not the main focus of our review.

TheMulti-Agent Programming Contest(https//multiagentcontest.org/)(MAPC) is an periodic transnational competition that occurs since 2005. Its purpose is to stimulate exploration inmulti-agent programming by introducing complex standard scripts that bear coordinated action and can be used to test and comparemulti-agent programming languages, platforms, and tools. executions using different agent- grounded platforms and languages have been used in the last many times; similar as JaCaMo(71 – 73), Jason(74,75), thing(76).

Agent- grounded models to pretend and estimate the transmission of the coronavirus complaint(COVID- 19) have been proposed in(77,78). There's an entire exploration area concentrated on using agent- grounded technologies in the energy assiduity. For illustration, in(79), MAS technologies are used for the control of Microgrid, its optimisation and request distribution. For farther reading, there's a check(80) on the operations of MAS in the control and operation of Microgrids, and a review(81) of the state of the art in the operation of MAS to energy optimisation problems.

In(82), an operation of ABMS is presented to study the connections between mortal conditioning and land- use/ land- cover changes to support scientific opinions regarding reasonable land planning and land use. The model is enforced grounded on the Repast modelling platform(83).

In(84), the authors present and illustrate FlowLogo, an interactive modelling terrain for developing coupled agent- grounded groundwater models. FlowLogois enforced in NetLogo and is the first intertwined software offering a straightforward way to represent agent behaviours that evolve with groundwater conditions. A methodical check on ABMS tools and operations can be set up in(8).

A methodological companion to the use of BDI agents in social simulations and an overview of being methodologies and tools for using them is handed in(10).

Agents can be used for developing tone-managed Internet of effects(IoT) systems due to their distributed nature, environment- mindfulness and tone- adaption(for farther readingabout using

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 8 Aug 2023, pp: 322-336 www.ijaem.net ISSN: 2395-5252

microservices as agents in IoT(85,86)). In(87) the authors aim to enhance the development of IoT operations using agents and software product lines in selfmanagement systems

In(88), trials to validate the programming of independent robots using Jadescript are presented. It presents the new support for perception instructors that has been lately introduced in the language to manage with the high data rate of detectors in robotic operations.

V. DISCUSSION AND FUTURE DIRECTION

As we've shown, there exists a wide variety of options for agent- grounded programming, from more traditional approaches(e.g., BDI) to simulation or planning-grounded. Some languages have also tried to combine generalities from agent- grounded programming with other programming paradigms, utmost prominently from object- acquainted programming. One of the main downsides of trying to achieve a wider community of programmers in agent- grounded programming is the lack of knowledge and familiarity with its generalities, that are significantly different from other more common paradigms. Agent- grounded programming languages that use some of the generalities from these other paradigms can help bring new programmers that would else be too bullied. These cold-blooded languages have their own niche of operations depending on the generalities that they use, which may occasionally imbrication with the further " pure " agent programming languages, but " pure " approaches are still necessary to completely attack all agent programming abstractions(agent, terrain, organisation, commerce,etc.).

Out of the 15 agent programming languages listed in Table 1, five don't have intimately available perpetration(i.e., the law isn't hosted in a public/ accessible sphere). This represents a significant issue, as it limits the practical usability of the proposed language and makes it delicate to quantitatively compare other languages against it. Not all extensions to being languages bear an perpetration to be useful, still, having one available is always positive for the community.

Indeed though there are several qualitative(e.g., generalities, features) comparisons in the literature, the use of different models of agency makes it delicate to give a fair comparison between the features present in these languages. A more in- depth study has to be conducted to identify the abecedarian features of agent acquainted programming, and more importantly, how these features fit in the different models of agency that being programming languages use.

Quantitative(e.g., performance) comparisons of programming languages are trickier due to the development cycle of having constant updates, which is indeed more common in programming languages developed in academia(as utmost of the agent programming languages are). nonetheless, it's important to develop agent-specific marks that can be used fluently by the community to estimate new programming languages or extensions to being languages.

Utmost languages offer a range of different exemplifications that showcase their features and strengths. While these exemplifications are clearly useful to more understand and learn the language, they're generally not enough to move new druggies of the connection of the language in real- world operations. Complex and realistic case studies are hard to develop, but the agent community has available a suite of complex scripts as part of the periodic MAPC that could be better exploited to test and compare agent programming languages.

Two recent checks(3,5) concentrated on BDI agent programming figure the limitations and challenges in the area. In a fiat(3), the author argues that it's necessary to extend the point set of current APLs to enable wider relinquishment of agent technology. The author also disagrees with once checks that the lack of further polished methodologies and tools isn't the main factor(although it does contribute to) in the limited relinquishment of APLs; rather, the author suggests that there's little to no incitement for inventors to make the change to AOP, as the behaviours presently shown in operations from the literature can be enforced in further mainstream languages with limited trouble. The check in(5) recaps the history so far and the state of the art in agent programming with a focus on BDI- grounded approaches. They identify as a major challenge for unborn exploration the integration of AI ways in agent programming languages as an important and necessary step to the wide acceptance and relinquishment of AOP.

Recommendation for Further Research

Considering the once 5 times of exploration on APLs, numerous new fabrics, platforms and models have been proposed. Each bone of these, along with new extensions, amended the agent- grounded literature and enlarged the diapason of possible operations.

nevertheless, as rightfully observed in(3,5), the major issue in current APLs isn't in their set of features, but in their usability. Generally, there's no desire in learning new languages when the advantages aren't straightforward. In our review, we analysed APLs that were both suggestive and important, but with major usability issues; similar as the absence of a (maintained) tool, attestation, and qualitative and quantitative comparisons with other languages. In our opinion, farther exploration on APLs will have to attack these usability issues in order to try to spread the use of APLs outside the agent community.

Agent- grounded programming is a thriving exploration area of artificial intelligence. In this review paper, we've classified both stager and recent benefactions according to four different orders agent- grounded programming languages, their extensions, the comparisons between languages, and eventually, some of the operations using these languages. For each donation we compactly reviewed the content and outlined the crucial results. To have a better understanding of the current state of art, we didn't only concentrate on the rearmost approaches, but we also compactly reviewed the most prominent agent-grounded programming languages that are still being maintained.

Every time there are numerous extensions to being languages and indeed entire new languages being proposed, still, utmost of them are limited to formal descriptions without any perpetration to support the formal proposition. The small subset of approaches with perpetration warrant any effective evaluation. Comparing new approaches to the state of the art is one of the major way needed to advance the area of agent-grounded programming. Qualitative and quantitative comparisons can help to identify gaps in being languages, which can lead to either advancements or new approaches that are suitable to manage with the challenges raised. also, in our review we've also linked a lack of real- world operations. In order to widen the use of these languages, it's important that their usability in the real- world is well proved, therefore, we encourage and recommend further operation-grounded papers that can demonstrate features of agent- grounded programming in the real- world

Funding

This exploration was funded by the UK Industrial Strategy Challenge Fund(ISCF) delivered by UK Research and Innovation(UKRI) and managed by Engineering and Physical lores Research Council(EPSRC) under the Robotics and AI for Extreme surroundings programme with subventions Robotics and AI in Nuclear(RAIN) mecca(EP/ R026084/ 1), unborn AI and Robotics for Space (FAIR- SPACE) mecca(EP/ R026092/ 1), and Offshore Robotics for Certification of means(ORCA) mecca (EP/ R026173/ 1).

Conflicts of Interest

The authors declare no conflict of interest. The funders had no part in the design of the study; in the collection, analyses, or interpretation of data; in the jotting of the handwriting, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AI Artificial Intelligence
AOP Agent-Oriented Programming
AOSE Agent-Oriented Software Engineering
APL Agent Programming Language
BDI Belief-Desire-Intention
CPS Cyber-Physical Systems
DSL Domain Specific Language
HTN Hierarchical Task Network
IoT Internet of Things
MAPC Multi-Agent Programming Contest
MAS Multi-Agent System
OOP Object-Oriented Programming
PRS Procedural Reasoning System

REFERENCES

[1]. Wooldridge, M. An preface to MultiAgent Systems, 2nded.; John Wiley and Sons Hoboken, NJ, USA, 2009; ISBN 047149691X.
[2]. Wooldridge,M.J.; Jennings,N.R. Intelligent agents proposition and practice. Knowl.Eng.Rev. 1995, 10, 115 – 152.(CrossRef)
[3]. Logan, B. An agent programming fiat.Int.J. Agent- acquainted Softw.Eng. 2018, 6, 187 – 210.(CrossRef)
[4]. Russell,S.J.; Norvig,P. Artificial Intelligence A Modern Approach, 3rded.; Prentice Hall Upper Saddle River, NJ, USA, 2010.
[5]. Bordini,R.H.; Seghrouchni,A.E.F.; Hindriks,K.V.; Logan,B.; Ricci,A. Agent programming in the cognitive period. Auton.Agents Multi Agent Syst. 2020, 34, 37.(CrossRef)
[6]. Mao,X.; Wang,Q.; Yang, S. A check of agent- acquainted programming from software engineering perspective. Web

Intell. 2017, , 143 – 163.(CrossRef) 7. Kravari,K.; Bassiliades, N. A check of Agent Platforms.J.Artif.Soc.Soc. Simul. 2015, 18.(CrossRef)

[7]. Abar,S.; Theodoropoulos,G.K.; Lemarinier,P.; O'Hare,G.M.P. Agent Based Modelling and Simulation tools A review of the state- of- art software. Comput.Sci.Rev. 2017, 24, 13 – 33.(CrossRef)

[8]. Isern,D.; Moreno, A. A methodical literature review of agents applied in healthcare.J. Med Syst. 2016, 40, 43.(CrossRef)

[9]. Adam,C.; Gaudou,B. BDI agents in social simulations a check. Knowl.Eng.Rev. 2016, 31, 207 – 238.(CrossRef)

[10]. Shoham,Y. Agent- acquainted Programming. Artif.Intell. 1993, 60, 51 – 92.(CrossRef)

[11]. Georgeff,M.; Lansky,A. Procedural Knowledge. Proc. IEEE(Spec. Issue Knowl. Represent.) 1986, 74, 1383 – 1398.(CrossRef)

[12]. Bratman,M.E. Intentions, Plans, and Practical Reason; Center for the Study of Language and Information Stanford, CA, USA, 1999.

[13]. Rao,A.S.; Georgeff,M. BDI Agents From proposition to Practice. In Proceedings of the First International Conference on Multiagent Systems(ICMAS), San Francisco, CA, USA, 12 – 14 June 1995;pp. 312 – 319.

[14]. Rao,A.S. AgentSpeak(L) BDI Agents Speak Out in a Logical Computable Language. In Agents Breaking Down, Proceedings of the th European Factory on Modelling Autonomous Agents in aMulti-Agent World, Eindhoven, The Netherlands, 22 – 25 January 1996; Lecture Notes in Computer Science; de Velde,W.V., Perram,J.W.,Eds.; Springer Berlin/ Heidelberg, Germany, 1996; Volume 1038, pp. 42 – 55.(CrossRef)

[15]. McCarthy,J.; Hayes,P.J. Some Philosophical Problems from the viewpoint of Artificial Intelligence. In Machine Intelligence 4; Meltzer,B., Michie,D.,Eds.; Edinburgh University Press Edinburgh, UK, 1969;pp. 463 – 502. distributed in McC90.

[16]. Issicaba,D.; Rosa,M.A.; Prostejovsky,A.M.; Bindner,H.W. Experimental confirmation of BDI agents for distributed control of electric power grids. In Proceedings of the 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe(ISGT- Europe), Torino, Italy, 26 – 29 September 2017;pp. 1 – 6.(CrossRef)

[17]. Sorici,A.; Boissier,O.; Picard,G.; Santi,A. Exploiting the JaCaMo Framework for Realising an Adaptive Room Governance operation. In Proceedings of the compendium of theCo-Located Workshops on DSM ' 11, TMC ' 11, AGERE! 2011, AOOPES ' 11, NEAT ' 11, and VMIL ' 11; New York, NY, USA, 1 – 31 October 2011;pp. 239 – 242.(CrossRef)

[18]. Persson,C.; Picard,G.; Ramparany,F.; Boissier, O. A JaCaMo- Grounded Governance of Machine- to- Machine Systems. In Advances on Practical operations of Agents andMulti-Agent Systems; Demazeau,Y., Müller,J.P., Rodríguez,J.M.C., Pérez,J.B.,Eds.; Springer Berlin/ Heidelberg, Germany, 2012;pp. 161 – 168.

[19]. Krupa,Y.; Vercouter,L. Handling sequestration as Contextual Integrity in Decentralized Virtual Communities The PrivaCIAS Framework. Web Intelli. Agent Syst. 2012, 10, 105 – 116.(CrossRef)

[20]. Collier,R.W.; Russell,S.E.; Lillis,D. Reflecting on Agent Programming with AgentSpeak(L). In Proceedings of the PRIMA 2015 Principles and Practice ofMulti-Agent Systems — 18th transnational Conference, Bertinoro, Italy, 26 – 30 October 2015; Lecture Notes in Computer Science; Chen,Q., Torroni,P., Villata,S., Hsu,J.Y., Omicini,A.,Eds.; Springer Cham, Switzerland, 2015; Volume 9387, pp. 351 – 366.(CrossRef)

[21]. Honorato- Zimmer,R.; Millar,A.J.; Plotkin,G.D.; Zardilis,A. Chromar, a language of parameterised agents. Theor.Comput.Sci. 2019, 765, 97 – 119.(CrossRef)

[22]. Hindriks,K.V.; de Boer,F.S.; van der Hoek,W.; Meyer,J.J.C. Agent Programming with Declarative pretensions. In Proceedings of the 7th International Workshop on Agent propositions, infrastructures, Boston, MA, USA, 7 – 9 July 2020;pp. 228 – 243.

[23]. Bordini,R.H.; Wooldridge,M.; Hübner,J.F. ProgrammingMulti-Agent Systems in

AgentSpeak Using Jason; John Wiley & Sons Hoboken, NJ, USA, 2007.
[24]. Boissier,O.; Bordini,R.H.; Hübner,J.F.; Ricci,A.; Santi,A.Multi-agent acquainted programming with JaCaMo. Sci. Comput. Program. 2013, 78, 747 – 761.(CrossRef)
[25]. Boissier,O.; Bordini,R.; Hubner,J.; Ricci,A.Multi-Agent acquainted Programming ProgrammingMulti-Agent Systems Using JaCaMo; Intelligent Robotics and Autonomous Agents Series; MIT Press Cambridge, MA, USA, 2020.
[26]. Ricci,A.; Piunti,M.; Viroli,M.; Omicini,A. Environment Programming in CArtAgO. InMulti-Agent Programming Languages, Tools and operations; Multiagent Systems, Artificial Societies, and Simulated Associations; Springer Boston, MA, USA, 2009; Chapter 8,pp. 259 – 288.(CrossRef)
[27]. Hübner,J.F.; Sichman,J.S.; Boissier,O. Developing organisedmultiagent systems using the MOISE model programming issues at the system and agent situations. Int.J. Agent- acquainted Softw.Eng. 2007, 1, 370 – 395.(CrossRef)
[28]. Dennis,L.A. Gwendolen Semantics 2017; Technical Report ULCS-17-001; University of Liverpool, Department of Computer Science Liverpool, UK, 2017. 30. Bellifemine,F.L.; Caire,G.; Greenwood,D. DevelopingMulti-Agent Systems with JADE(Wiley Series in Agent Technology); John Wiley & Sons Hoboken, NJ, USA, 2007.
[29]. Bergenti,F.; Iotti,E.; Monica,S.; Poggi,A. Agent- acquainted model- driven development for JADE with the JADEL programming language. Comput. Lang. Syst. Struct. 2017, 50, 142 – 158.(CrossRef)
[30]. Bergenti,F.; Monica,S.; Petrosino, G. A scripting language for practical agent-acquainted programming. In Proceedings of the 8th ACM SIGPLAN International Workshop on Programming Grounded on Actors, Agents, and Decentralized Control, AGERE!@SPLASH 2018, Boston, MA, USA, 5 November 2018;pp. 62 – 71.(CrossRef)
[31]. 33. Pokahr,A.; Braubach,L.; Lamersdorf,W., Jadex A BDI logic Machine. InMulti-Agent Programming Languages, Platforms and operations; Springer Boston, MA, USA, 2005;pp. 149 – 174.(CrossRef)
[32]. Aschermann,M.; Dennisen,S.; Kraus,P.; Müller,J.P. LightJason, a largely Scalable and Concurrent Agent Framework Overview and operation. In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS , Stockholm, Sweden, 10 – 15 July 2018;pp. 1794 – 1796.
[33]. Hashmi,M.A.; Seghrouchni,A.E.F.; Akram,M.U. A Planning Grounded Agent Programming Language Supporting Environment Modeling. In Proceedings of the IEEE/ WIC/ ACM International Conference on Web Intelligence and Intelligent Agent Technology, WI- IAT 2015, Singapore, 6 – 9 December 2015;pp. 76 – 83.(CrossRef)
[34]. Kilaru,J. PLASA Programming Language for Synchronous Agents. Master's Thesis, California State University, Long Beach, CA, USA, 2018.
[35]. Flocchini,P.; Prencipe,G.; Santoro,N.; Widmayer,P. Gathering of asynchronous robots with limited visibility. Theor.Comput.Sci. 2005, 337, 147 – 168.(CrossRef)
[36]. Bonci,A.; Pirani,M.; Bianconi,C.; Longhi,S. RMAS Relational Multiagent System for CPS Prototyping and Programming. In Proceedings of the 14th IEEE/ ASME International Conference on Mechatronic and Bedded Systems and Applications, MESA 2018, Oulu, Finland, 2 – 4 July 2018;pp. 1 – 6.(CrossRef)
[37]. Rodriguez,S.; Gaud,N.; Galland,S. SARL A General- Purpose Agent- acquainted Programming Language. In Proceedings of the 2014 IEEE/ WIC/ ACM International Joint Conferences on Web Intelligence(WI) and Intelligent Agent Technologies(IAT), Warsaw, Poland, 11 – 14 August 2014; Volume III,pp. 103 – 110;(CrossRef)
[38]. Molesini,A.; Casadei,M.; Omicini,A.; Viroli,M. Simulation in agent- acquainted software engineering The SODA case study.Sci. Comput.Program. 2013, 78, 705 – 714.(CrossRef)
[39]. García- Magariño,I.; Gómez-Rodríguez,A.; Moreno,J.C.G.; Navarro,G.P. PEABS A Process for developing Effective Agent- Grounded Simulators.Eng. Appl. Artif. Intell. 2015, 46, 104 – 112.(CrossRef)

[40]. Pavón,J.; Gómez- Sanz,J.; Fuentes-Fernández,R., The INGENIAS methodology and tools. In Agent-acquainted Methodol; IGI Global Hershey, PA, USA, 2005;pp. 236 – 276.(CrossRef)

[41]. Caillou,P.; Gaudou,B.; Grignard,A.; Truong,Q.C.; Taillandier, P. A Simple- to-Use BDI Architecture for Agent-Grounded Modeling and Simulation. In Proceedings of the European Social Simulation Association 2015, Groningen, The Netherlands, – 18 September 2015; Volume 528,pp. 15 – 28.(CrossRef) 44. Taillandier,P.; Bourgais,M.; Caillou,P.; Adam,C.; Gaudou, B. A BDI Agent Architecture for the GAMA Modeling and Simulation Platform. In Proceedings of theMulti-Agent Grounded Simulation XVII — International Workshop, MABS 2016, Singapore, 10 May ; Volume 10399,pp. 3 – 23.(CrossRef)

[42]. Grignard,A.; Taillandier,P.; Gaudou,B.; Vo,D.; Huynh,N.Q.; Drogoul,A. GAMA1.6 Advancing the Art of Complex AgentBased Modeling and Simulation. In Proceedings of the PRIMA 2013 Principles and Practice ofMulti-Agent Systems — 16th transnational Conference, Dunedin, New Zealand, 1 – 6 December 2013; Volume 8291,pp. 117 – 131.(CrossRef)

[43]. Singh,D.; Padgham,L.; Logan,B. Integrating BDI Agents with Agent-Grounded Simulation Platforms. Auton. Agents Multi Agent Syst. 2016, 30, 1050 – 1071.(CrossRef)

[44]. Belle,V.; Levesque,H.J. PREGO An Action Language for Belief- Grounded Cognitive Robotics in nonstop disciplines. In Proceedings of the Twenty- Eighth AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27 – 31 July 2014;pp. 989 – 995.

[45]. Belle,V.; Levesque,H.J. ALLEGRO Belief- Grounded Programming in Stochastic Dynamical disciplines. In Proceedings of the Twenty- Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, 25 – 31 July 2015; pp. 2762 – 2769.

[46]. Levesque,H.J.; Reiter,R.; Lespérance,Y.; Lin,F.; Scherl,R.B. GOLOG A Logic Programming Language for Dynamic disciplines.J. Log. Program. 1997, 31, 59 – 83.(CrossRef)

[47]. Ferrein,A.; Maier,C.; Mühlbacher,C.; Niemueller,T.; Steinbauer,G.; Vassos,S. Controlling Logistics Robots with the Action- Grounded Language YAGI. In Proceedings of the Intelligent Robotics and Applications — 9th transnational Conference, ICIRA 2016, Tokyo, Japan, 22 – 24 August 2016; Volume 9834,pp. 525 – 537.(CrossRef)

[48]. Quigley,M.; Conley,K.; Gerkey,B.; Faust,J.; Foote,T.; Leibs,J.; Wheeler,R.; Ng,A. ROS An open- source Robot Operating System. In Proceedings of the Factory on Open Source Software at the International Conference on Robotics and robotization, Kobe, Japan, 12 – 13 May 2009;p. 5.

[49]. Kaptein,F.; Broekens,J.; Hindriks,K.V.; Neerincx,M.A. CAAF A Cognitive Affective Agent Programming Framework. In Proceedings of the Intelligent Virtual Agents 16th International Conference, IVA 2016, Los Angeles, CA, USA, 20 – 23 September ; Volume 10011,pp. 317 – 330.(CrossRef)

[50]. Praça,I.; Ramos,C.; Vale,Z.; Cordeiro,M. MASCEM A multiagent system that simulates competitive electricity requests. IEEE Intell. Syst. 2003, 18, 54 – 60.(CrossRef)

[51]. Santos,G.; Pinto,T.; Praça,I.; Vale,Z. MASCEM Optimizing the performance of amulti-agent system. Energy 2016, 111, 513 – 524. CrossRef)

[52]. García- Magariño,I.; Navarro,G.P.; Lacuesta,R. TABSAOND A fashion for developing agent- grounded simulation apps and online tools with nondeterministic opinions. Simul.Model.Pract.proposition 2017, 77, 84 – 107.(CrossRef)

[53]. Cich,G.; Galland,S.; Knapen,L.; Yasar,A.; Bellemans,T.; Janssens,D. Addressing the Challenges of Conservative Event Synchronization for the SARL Agent-Programming Language. In Proceedings of the Advances in Practical operations of Cyber-PhysicalMulti-Agent Systems, PAAMS Collection — 15th International Conference, PAAMS 2017, Porto, Portugal, 21 – 23 June 2017; Volume 10349,pp. 31 – 42.(CrossRef)

[54]. Jain,S.; Asawa,K. Programming an suggestive independent agent. Expert

Syst. Appl. 2016, 43, 131 – 141.(CrossRef)

[55]. Jain,S.; Asawa,K. EMIA emotion model for intelligent agent.J. Intell. Syst. 2015, 24, 449 – 465.(CrossRef)

[56]. Dastani,M. 2APL A practical agent programming language. Auton.AgentsMulti-Agent Syst. 2008, 16, 214 – 248.(CrossRef)

[57]. Pantoja,C.E.; Stabile,M.F.; Lazarin,N.M.; Sichman,J.S. ARGO An Extended Jason Architecture that Facilitates Bedded Robotic Agents Programming. In Proceedings of the EngineeringMulti-Agent Systems — 4th transnational Factory, EMAS , Singapore, 9 – 10 May 2016; Volume 10093,pp. 136 – 155.(CrossRef)

[58]. Leask,S.; Logan,B. Programming deliberation strategies in meta- APL. In Proceedings of the International Conference on Principles and Practice ofMulti-Agent Systems, Bertinoro, Italy, 26 – 30 October 2015;pp. 433 – 448.

[59]. Doan,T.T.; Yao,Y.; Alechina,N.; Logan,B. Verifying miscellaneousmulti-agent programs. In Proceedings of the International conference on Autonomous Agents andMulti-Agent Systems, AAMAS ' 14, Paris, France, 5 – 9 May 2014;pp. 149 – 156.

[60]. Cardoso,R.C.; Ferrando,A.; Dennis,L.A.; Fisher, M. An Interface for Programming Verifiable Autonomous Agents in ROS. In Proceedings of the European Conference onMulti-Agent Systems(EUMAS), Thessaloniki, Greece, 14 – 15 September 2020.

[61]. Onyedinma,C.; Gavigan,P.; Esfandiari,B. Toward Campus Mail Delivery Using BDI.J.Sens. Actuator Netw.2020, 9, 56.CrossRef)

[62]. Bosello,M.; Ricci, A. From Programming Agents to Educating Agents- A Jason-Grounded Framework for Integrating Learning in the Development of Cognitive Agents. In Proceedings of the EngineeringMulti-Agent Systems — 7th International Workshop, EMAS , Montreal, QC, Canada, 13 – 14 May 2019; Volume 12058,pp. 175 – 194.(CrossRef)

[63]. Cardoso,R.C.; Zatelli,M.R.; Hübner,J.F.; Bordini,R.H. Towards Benchmarking Actor- and Agent- Based Programming Languages. In Proceedings of the Factory on Programming Grounded on Actors, Agents, and Decentralized Control, Indianapolis, IN, USA, October 2013;pp. 115 – 126.

[64]. Challenger,M.; Kardas,G.; Tekinerdogan, B. A methodical approach to assessing sphere-specific modeling language surroundings formulti-agent systems. Softw.Qual.J. 2016, 24, 755 – 795.(CrossRef)

[65]. Bergenti,F.; Iotti,E.; Monica,S.; Poggi, A. A Comparison between Asynchronous Backtracking Pseudocode and its JADEL perpetration. In Proceedings of the 9th International Conference on Agents and Artificial Intelligence, ICAART, Porto, Portugal, 24 – 26 February 2017; Volume 2,pp. 250 – 258.(CrossRef) 69. Rousset,A.; Herrmann,B.; Lang,C.; Philippe, L. A check on resemblant and distributedmulti-agent systems for high performance computing simulations. Comput.Sci.Rev. 2016, 22, 27 – 46.(CrossRef)

[66]. Sun,Z.; Lorscheid,I.; Millington,J.D.A.; Lauf,S.; Magliocca,N.R.; Groeneveld,J.; Balbi,S.; Nolzen,H.; Müller,B.; Schulze,J.; etal. Simple or complicated agent-grounded models? A complicated issue. Environ. Model.Softw. 2016, 86, 56 – 67.(CrossRef)

[67]. Cardoso,R.C.; Krausburg,T.; Baségio,T.L.; Engelmann,D.C.; Hübner,J.F.; Bordini,R.H. SMART-JaCaMo An association- grounded platoon for themulti-agent programming contest. Ann. Math. Artif.Intell. 2018, 84, 75 – 93.(CrossRef)

[68]. Krausburg,T.; Cardoso,R.C.; Damasio,J.; Peres,V.; Farias,G.P.; Engelmann,D.C.; Hübner,J.F.; Bordini,R.H. SMART – JaCaMo An Organisation- Grounded platoon for theMulti-Agent Programming Contest. In TheMulti-Agent Programming Contest 2018; Ahlbrecht, T., Dix,J., Fiekas,N.,Eds.; Springer International Publishing Cham, Switzerland, 2019;pp. 72 – 100.

[69]. Cardoso,R.C.; Ferrando,A.; Papacchini,F. LFC Combining Autonomous Agents and Automated Planning in theMulti-Agent Programming Contest. InMulti-Agent Progamming Contest; Springer Cham, Switzerland, 2019;pp. 31 – 58.

[70]. Vezina,M.; Esfandiari, B. The demand Gatherers ' Approach to the 2019Multi-Agent Programming Contest script. In The

Multi-Agent Programming Contest 2019; Ahlbrecht,T., Dix,J., Fiekas,N., Krausburg,T.,Eds.; Springer International Publishing Cham, Switzerland, 2020;pp. 106 – 150.

[71]. Villadsen,J.; Bjørn,M.O.; From,A.H.; Henney,T.S.; Larsen,J.B.Multi-Agent Programming Contest 2018 — The Jason-DTU Team. In TheMulti-Agent Programming Contest 2018; Ahlbrecht,T., Dix,J., Fiekas,N.,Eds.; Springer International Publishing Cham, Switzerland, 2019;pp. 41 – 71.

[72]. Jensen,A.B.; Villadsen,J. GOAL- DTU Development of Distributed Intelligence for theMulti-Agent Programming Contest. In TheMulti-Agent Programming Contest 2019; Ahlbrecht,T., Dix,J., Fiekas,N., Krausburg,T.,Eds.; Springer International Publishing Cham, Switzerland, 2020;pp. 79 – 105.

[73]. Wolfram, C. An Agent- Grounded Model of COVID- 19. Complex Syst. 2020, 29.(CrossRef)

[74]. Prudhomme,C.; Cruz,C.; Cherifi, H. An Agent grounded model for the transmission and control of the COVID-19 in Dijon(extended abstract). In Proceedings of MARAMI 2020 Modèles&Analyse des RéseauxApprochesMathématiques&Informatiques — The 11th Conference on Network Modeling and Analysis, Virtual Conference, Montpellier, France, 14 – 15 October 2020; Volume 2750.

[75]. Khan,M.W.; Wang, J. The explorationonmulti-agent system for microgrid control and optimization. Renew. Sustain. EnergyRev. 2017, 80, 1399 – 1411.(CrossRef)

[76]. Kantamneni,A.; Brown,L.E.; Parker,G.G.; Weaver,W.W. Survey ofmulti-agent systems for microgridcontrol.Eng. Appl. Artif. Intell. 2015, 45, 192 – 203.(CrossRef)

[77]. González- Briones,A.; De La Prieta,F.; Mohamad,M.S.; Omatu,S.; Corchado,J.M.Multi-agent systems operations in energy optimization problems A state- of- the- art review. powers 2018, 11, 1928.(CrossRef)

[78]. QuanLi,X.; Kun,Y.; GuiLin,W.; YuLian,Y. Agent- grounded modeling and simulations of land- use and land- cover change according to ant colony optimization A case study of the Erhai Lake Basin, China. Nat. Hazards 2015, 75, 95 – 118.(CrossRef)

[79]. North,M.; Collier,N.; Ozik,J.; Tatara,E.; Macal,C.; Bragen,M.; Sydelko,P. Complex Adaptive Systems Modeling with Repast Simphony. Complex Adapt. Syst. Model. 2013, 1, 1 – 26.(CrossRef)

[80]. Castilla- Rho,J.C.; Mariethoz,G.; Rojas-Mujica,R.; Andersen,M.S.; Kelly,B.F.J. An agent- grounded platform for bluffing complex mortal- aquifer relations in managed groundwater systems. Environ. Model.Softw. 2015, 73, 305 – 323.(CrossRef)

[81]. Savaglio,C.; Fortino,G.; Ganzha,M.; Paprzycki,M.; Badica,C.; Ivanovic,M. Agent- Based Computing in the Internet of effects A Survey. In Proceedings of the Intelligent Distributed Computing XI — 11th transnational Symposium on Intelligent Distributed Computing — IDC 2017, Belgrade, Serbia, 11 – 13 October 2017; Volume 737,pp. 307 – 320.(CrossRef)

[82]. Krivic,P.; Skocir,P.; Kusek,M.; Jezic,G. Microservices as Agents in IoT Systems. In Proceedings of the Agent andMulti-Agent Systems Technology and Applications, 11th KES International Conference, KES- AMSTA 2017, Vilamoura, Algarve, Portugal, – 23 June 2017; Volume 74,pp. 22 – 31.(CrossRef)

[83]. Ayala,I.; Amor,M.; Fuentes,L.; Troya,J.M. A Software Product Line Process to Develop Agents for the IoT. Detectors 2015, , 15640 – 15660.(CrossRef) 88. Iotti,E.; Petrosino,G.; Monica,S.; Bergenti,F. Exploratory trials on Programming Autonomous Robots in Jadescript. In Proceedings of the First Workshop on Agents and Robots for dependable Engineered Autonomy,AREA@ECAI 2020, Virtual Event, Santiago de Compostela, Spain, 4 September 2020; Volume 319,pp. 55 – 67.(CrossRef)