Agent-Based Computing in the Internet of Things: a Survey

Claudio Savaglio [1], Giancarlo Fortino [1], Maria Ganzha [2], Marcin Paprzycki [3], Costin Bădică [4], and Mirjana Ivanović [5]

[1] DIMES, Università della Calabria, Rende (CS), Italy
[2] Warsaw University of Technology, Warsaw, Poland
[3] Systems Research Institute Polish Academy of Sciences, Warsaw, Poland
[4] CIT, University of Craiova, Craiova, Romania
[5] DMI, University of Novi Sad, Serbia
csavaglio@dimes.unical.it,g.fortino@unical.it,m.ganzha@mini.pw.edu.pl,
marcin.paprzycki@ibspan.waw.pl,cbadica@software.ucv.ro,mira@dmi.uns.ac.rs

Abstract. The Internet of Things is a revolutionary concept, within cyberphysical systems, rich in potential as well as in multifacet requirements and development issues. To properly address them and to fully support IoT systems development, Agent-Based Computing represents a suitable and effective modeling, programming, simulation paradigm. As matter of facts, agent metaphors, concepts, techniques, methods and tools have been widely exploited to develop IoT systems. Main contemporary contributions in this direction are surveyed and reported in this work.

Keywords: Internet of Things, Agent-based Computing, Modeling, Architectures, Simulation, Methodology

1 Introduction

Since early 2000s, technological advances in wireless communication, embedded processing, sensing and actuation, are fueling rapid spread of novel cyberphysical artifacts. Ranging from simple movement detectors and temperature sensors, to more sophisticated smartphones and smart cars, they can sense the physical world, process data, and impact the surrounding environment in different ways, for example by triggering actions through actuators or engaging customized users interactions. In the context of the Internet of Things (IoT) [1], such devices have been massively networked and provided with (different degrees of) intelligence, being defined as "Smart Objects" (SOs) [2]. They communicate with each other, as well as with conventional computing systems, and cooperate in a synergic fashion, both on local and global scales, to implement cyberphysical applications in multitude of scenarios, e.g., industrial automation, logistic optimization, energy management, public security, entertainment, ambient assisted living and wellness, to name just a few.

To comprehensively support needs arising in complex development of heterogeneous IoT applications and systems, different mainstream paradigms and approaches (especially in closely related fields of wireless sensor networks, distributed systems,

ubiquitous and pervasive computing), have been jointly exploited [3]. Among these, Agent-based Computing (ABC) [4] has been widely recognized as full-fledged, effective support for development of decentralized, dynamic, cooperating and open IoT systems, particularly in conjunction with other complementary paradigms, e.g., cloud [5] and autonomic [6] computing.

In this paper, our intention is to show how ABC has been effectively exploited for modeling, programming and simulating IoT systems. Indeed, the ABC provides ideas, metaphors, techniques, methods and tools for systematically conceptualizing, realizing and simulating distributed systems composed of heterogeneous interacting entities [7]. Therefore, in Section 2, we first provide some insights specifically focused on the main IoT development issues and distinctive features of ABC. In Section 3, we survey several contributions exploiting ABC in the IoT context, for modeling, programming and simulation purposes. Ultimately, a brief analysis of surveyed state-of-the-art and some final remarks conclude the work.

2 Background

2.1 Internet of Things Development Challenges

The Internet of Things consists of great number and variety of components (RFiD, sensors, conventional laptops, micro and super computers, smartphones, robots, home appliances, vehicles, etc.), network types (Bluetooth-based personal area networks, ZigBee-based industrial networks, 5G and Wifi-based very dense metropolitan area networks, etc.), and stakeholders (citizens, private companies, public administrations, other digital systems, etc.), thus constituting an extremely multi-facet global ecosystem [1]. Because of heterogeneity of IoT building blocks, lack of standards, massive scale (the total number of "things" is forecasted to reach 20.4 billion in 2020) and rapid evolution, development of IoT applications and systems involves large number of requirements and issues [3]. In particular, IoT devices, also denoted as Smart Objects (SOs, in contrast with simple resources as sensors, actuators, databases, etc.) are expected not only to be intelligent, context-aware and autonomous, but also easy to use, reliable and secure. The same desiderata should apply to all IoT systems that are expected to be autonomic, scalable and open, thus avoiding the proliferation of poorly interoperable "intra-nets of things" [6]. Beside the fulfillment of such requirements, unexpected issues related to different development phases need to be considered and dynamically handled (e.g., an already deployed application needs to expose new functionality and interfaces for interacting with novel SOs). In this context, a multidisciplinary and systematic approach, involving different expertise for coping with the cyberphysical nature of IoT ecosystem [8], is necessary. Henceforth, full-fledged IoT methodologies are gaining traction [9], aiming at systematically supporting all development phases, addressing mentioned issues, and reducing time-to-market, efforts and probability of failure.

2.2 Agent-Based Computing Paradigm

Agent-based Computing is centered around the concept of an agent [4], a sophisticated software abstraction defining an autonomous, social, reactive and proactive entity. Agents are situated in some environment (namely, world of perceived resources)

and act to achieve their design objectives, exhibiting flexible problem solving behaviors. Agents, interacting and cooperating to solve/realize problems/services that are beyond the capabilities of a single agent, constitute a MAS (Multi Agent System) [4]. MASs are distributed and self-steering societies, featured by a strong situatedness and well-defined organizational relationships, covering variety of domains (e.g., sociology, economy, logistics). The above characterization, although not exhaustive, indicates that ABC provides a set of key abstractions and metaphors for straightforwardly *modeling* complex systems, their components, interactions and organizational relationships.

Beside modeling, ABC is also a well-established *programming* paradigm for concretely implementing agents' advanced features, and effectively addressing key requirements typical of modern (distributed) applications. Indeed, agent's, society's and environment's modeling abstractions have been exploited to devise a high-level, distributed programming paradigm, centered around two cornerstones [10]: (1) encapsulation of control (that consists in giving each agent its own thread of control and reasoning capabilities, thus designing context-aware entities with autonomous behaviors), and (2) interaction (including coordination and cooperation mechanisms, based on high-level asynchronous message passing). Here, adoption of shared communication standards and management specifications (e.g., the IEEE FIPA-based system platforms and communication languages [11, 12]) allows agents to act also as interoperability facilitators, by incorporating within the agent society a variety of resources and existing legacy systems. Such advantages enable agent-based programming paradigm to enhance system's performance (i.e, computational efficiency, reliability, responsiveness, etc.), interoperability and scalability, specially with respect to the centralized approaches.

Finally, computing systems, modeled and programmed following the agent-oriented approach, can be straightforwardly simulated, for effectively studying macro phenomena and patterns, as well as individual behaviors and environment evolution [13]. Indeed, agent-based *simulation* allows evaluating agent-based systems exposing discrete, not linear, adaptive behaviors even in highly interacting, distributed, scaling-up, virtual scenarios. To properly exploit the surveyed agent-oriented metaphors, techniques and tools, thus providing a systematical approach to the agent-based modeling, programming, and simulation, several agent-oriented development *methodologies* have been designed and successfully applied [14]. However, as highlighted in [15] and [16], ABC is neither a universal nor necessarily effective development solution, since agent-level and society-level pitfalls can occur from different perspectives (management, conceptual, design, etc.), thus outweighing any agent-related benefits. Therefore, the adoption of ABC paradigm needs to be carefully assessed.

3 Agents' Contribution in Developing IoT Systems

The agent-oriented view of the world is perhaps the most natural way of approaching several types of (natural and artificial) systems, featured by a relevant complexity, dynamicity, situatedness and autonomy [7]. In particular, strong conceptual relation exists between agents and SOs, as well as between MAS and IoT systems [43]. Thus, considering the entire set of requirements and issues related to the development of IoT systems, ABC has been exploited for modeling, programming and simulating IoT applica-

tions and systems, and thus systematically driving and speeding-up their development. The most relevant contributions which exploit ABC for these purposes have been surveyed and compared, according to their provided agent-based features in Tab. 1, summarizing the outcomes of subsections 3.1-3.4. In detail, for each contribution, Tab. 1 indicates if it performs a fine or coarse grained agent-based IoT entity modeling, if it implements mechanisms for (technological/syntactical/semantic) interoperability, autonomicity, cognitivity, virtualization or security, if (and how) it performs IoT system simulation, and finally, if it presents an agent-based IoT development methodology.

Table 1. Surveyed works and provided agent-based features - *T=technological, Sy=syntactical, Se=semantic interoperability; A=autonomicity; C=cognitivity; V=virtualization; S=security.*

Surveyed work <name, ref. >	**Agent-based IoT model**		**Agent-based IoT implementation**							**Agent-based IoT simulation**		**Agent-based IoT Methodology**
	Fine grained	*Coarse grained*	*T*	*Sy*	*Se*	*C*	*A*	*V*	*S*	*Pure*	*Hybrid*	
Cascadas, [17]	X		X	X			X					
iCore, [21]	X		X	X	X	X			X			
ACOSO [6], [9], [22], [42], [43]	X		X	X	X	X	X	X			X	X
UBIWARE, [19]; UBIROAD, [26]	X		X	X	X	X		X	X			
[29]		X	X	X				X				
[44]	X		X	X	X		X	X				X
AoT, [27]		X	X	X		X	X	X				
Smart Grids, [40]		X		X						X		
[41]		X	X	X				X			X	
TAEC, [39]		X	X	X	X		X	X	X			
CIoT, [32]	X		X	X	X	X		X	X			
iSapiens, [23]	X		X	X				X	X			
[31]		X	X	X				X				
Radigost, [38]		X	X	X				X				
ASSIST, [45]	X		X	X	X	X		X		X		
BEMOSS, [25]		X	X	X				X	X			
INTER-IoT, [50]	X		X	X	X	X	X	X	X		X	X
VICINITY, [33]	X		X	X	X	X		X	X			
SOL, [28]	X		X	X	X		X					
[20]	X		X	X	X							
[24]		X	X	X			X					
[30]		X	X	X		X	X					
Smart Santander, [35]		X	X	X	X			X	X			
[34]		X	X	X	X			X	X			
Prometheus, [46]	X		X	X	X			X	X			X
ASEME, [18]	X		X	X	X			X	X			X
SAMSON, [51]	X						X			X		

3.1 ABC as IoT Modeling Paradigm

Agent-based modeling allows capturing key characteristics of SOs and IoT systems, at different degrees of granularity and in a technology-agnostic way. Indeed, SO autonomicity, proactiveness and situatedness are implicitly embedded in the agent model, while other important SO features can be explicitly described through agent-related concepts. This is the case of [17, 24], [51], which express SO functionalities in terms of goals, SO working plan in terms of behaviors, and SO augmentation-related components (like knowledge bases, sensors and actuators) in terms of dynamically bindable agent resources. However, these works adopt different mechanisms for specifically characterizing SOs/agents. In particular, in [18, 20] each agent/SO has a role (taken from a scenario-dependent repository, e.g., smart car, smart driver-support or smart road for the transportation context) that determines, by default, its own behaviors, goals and communications paradigms; similarly, in [21], SO/agent plans and goals are encoded in templates reflecting their functionalities. Other contributions do not reference a-priori defined roles or templates. For example, in [17], each agent/SO has a self-model (an automaton) driving its actions according to stimuli (modeled as messages) from other agents or the environment. Similarly, in [22, 23], SOs' actions/reactions are encoded in behaviors, driven by incoming (internal/external) events and design goals (encapsulated in state-based tasks). Finally, in [24], SO/agent self-state is dynamically determined by combining its real-time sensor data, position and status of computational units.

Furthermore, surveyed agent-oriented SO models are particularly suitable for supporting preliminary development phase of analysis, abstracting main SOs features from low-level details or specific implementation constraints. Beside the satisfactory "perse" agent-based SOs descriptions, however, further research efforts are necessary to thoughtfully model relationships among cyberphysical agents/SOs interacting within physical everyday environments. Nevertheless, agent-based IoT models represent a convenient starting point for subsequent phases of agent-oriented programming and simulations [9].

3.2 ABC as IoT Programming Paradigm

Because of deep heterogeneity of resources and communication protocols in the IoT context, many authors propose an agent-oriented approach for programming uniform interfaces and thus transparently interacting with resources and SOs. Authors of [19], [21, 22], [26], [29], exploit software adapters (developed for specific technologies and coordinated internally by a device manager [22]) for accessing agent/SO augmentation devices. This approach improves modularity and extendibility, since it leverages pluggable software components that can be defined when needed, and customized within the target resource. Instead, [20], [25], follow a different approach: each resource is directly coupled with one agent that interfaces the resource itself with the related SO, or with rest of the system. This solution completely hides the underlying technological heterogeneity, but it is not suitable for such constrained devices that cannot support an agent-based architecture. Apart from resource handling, agent-oriented programming contributes to overcoming lack of communication/coordination standards within the IoT arena: (i) by

implementing the IEEE FIPA 'de facto' standard specifications [11], and (ii) by supporting the SOs virtualization [21], thus paving the way towards integration of the agentified SOs within the Cloud [5], [23] (outsourcing computation/storage, and thus mitigating the SO hardware/software limitations) and with Web Services [27] (thus enhancing the SOs accessibility). Indeed, FIPA specifications standardize message format (specifically, the Agent Communication Language [12], ACL, is used for encoding message envelope) and message content (whose concepts, typically expressed through metadata-oriented languages, refer to ontology for facilitating data and the context management), and provide effective message transport service (leveraging on both semi/centralized and distributed services of agent discovery). Conversely, SOA and REST make SOs functionalities accessible under the form of Web Services over standard Internet protocols that are platforms and programming languages independent [31], [38].

Summarizing, agents are powerful mechanisms that realize the following functions:

- technical interoperability through shared resource/communication interfaces, as emphasized in [17], [22],[28], [30], [35], [38] (however, agents developed by different organizations are unable to interoperate well, as FIPA standard arent ready to support full interoperability in real-time/distributed control and diagnostic [47]);
- syntactical inteoperability through a shared message format, because ACL is adopted across FIPA standard obeying platforms for message envelope, while XML and JSON are used for message content in [19, 20], [29], [31], [33], [35] (but it is worth noting that ACL is a 'de facto' standard and other languages, like Knowledge Query and Manipulation Language, KQML, have some success [15]); and
- semantic interoperability through shared ontology and knowledge representation, as particularly done in [19, 21], [26], [32], [34, 36] (although this function is quite limited and underveloped due to the scarcity of grounded domain-specific ontology and semantics [16]).

At a higher level of conceptualization, agents allow to straightforwardly instill smartness and autonomy within a single SO, and realize cognitive and autonomic IoT systems [6], [37]. In fact, agent-based programming paradigm enables development of (i) self-configuring, self-healing, self-protecting and self-optimizing SOs/IoT systems, that are manageable with a minimum human intervention [17], [28], [30], [39]; and (ii) self-learning, context-aware and adaptive SOs/IoT systems [21], [32, 33], capable of solving problems without requiring human assistance. Autonomic and cognitive features are particularly important for ensuring self-management, distributed intelligence and scalability, but also in the perspective a secured and trusted IoT scenario, supporting the implementation of conventional, as well as unconventional, trust mechanisms. The first case refers to certificate-based reputation systems [21], [39]; the second one refers to the Social IoT approach [23], [25], [33], [45] that leverages on the inter-SOs relationships (e.g., location, ownership, chronology of mutual interactions).

3.3 ABC as IoT Simulation Paradigm

Particularly in the IoT context, where interactions are subject to variety of contingent factors [51] (e.g., SOs density, physical network design, traffic congestion, wireless

signal attenuation and coverage), and deployment is often error prone and time consuming, being able to simulate the system plays a crucial role [42, 43]. In fact, it allows understanding overall dynamics, estimating performance, and validating models, protocols and algorithms featuring under-development SOs and IoT systems. Although agent-based simulators allow effectively inspecting high-level aspects such as the raise of collective dynamics and behavioral patterns, they typically neglect, or coarsely handle, low-level communication issues, thus resulting in quasi aseptic simulation environments that are far from the actual cyberphysical IoT scenarios [40], [45]. Therefore, authors in [6], [41, 43], propose an hybrid approach, based on joint exploitation of agent-oriented modeling and network-based simulation. Such complementary approach allows mitigating limitations of pure agent-based simulation (but maintaining advantages derived from ABC) and effectively simulating IoT systems of different scales (from small ad-hoc networks, to large and dense Smart Cities) with variety of configurations (different communication patterns, protocols, parameters) [6], [42, 43].

Differently from well-established agent-based modeling and programming paradigm, research in agent-based IoT simulation is in its infancy. However, considering that currently IoT-specific simulators are not available, the hybrid agent-based approach represents one of the few state-of-the-art candidates for supporting the crucial activity of IoT system simulation. Overlooking such aspect could be a critical pitfall that compromises the agents acceptance in the IoT context, since MASs have no central control and thus unpredictable and emergent behaviors are likely [47].

3.4 Agent-Based Methodology for IoT

Some agent-oriented methodologies have been specifically extended for the IoT context [18], [46], or ex-novo designed [9], [44]. However, beside disciplining the exploitation of agent-based suite of models, programming techniques and simulation tools [46], requirements that are typically overlooked by agent-based methodologies need to be considered. In this direction, to support the IoT system development in all its phases, the aforementioned agent-based methodologies:

- thoroughly consider the cyberphysical nature of the involved entities and environments, foreseeing by design, solutions for interoperability, security and scalability [9], [44], [46];
- emphasize identification of IoT users and stakeholders, depicting significant use cases through textual descriptions [44] and technical notations, like UML (Unified Modeling Language) [18] or BPMN (Business Process Model Notation) [46], for meeting different expertise and perspectives;
- define the proper management, coordination and virtualization mechanisms [9], typically situating them at the middleware level [48, 49], for gluing hardware and software components;
- analyze infrastructural features and limitations according to the specific IoT system requirements [18], [44], since these factors cannot be considered independently;
- provide guidelines and best-practices for unbinding developers from a specific technology or protocol, driving and promoting integration of different computing paradigms and application contexts [9], [50].

Without extensively dealing with all these factors, even effective and well-known conventional agent-based software development methodologies like Tropos [52] are definitively inadequate, and unable to actually unfold the full IoT potential.

4 Analysis and Concluding Remarks

IoT full realization is not hindered by hardware constraints or computational / storage / communication limitations, but by some requirements that have not been totally or simultaneously addressed. Leveraging agents key features of autonomy, proactiveness, intelligence and sociability and, according to the numerous contributions surveyed in this work, we believe that ABC can be effectively exploited as modeling, programming, and simulation paradigm for developing IoT ecosystems. Indeed, better than other computing paradigms (object-oriented, service-oriented, component-oriented) and both at things and at system levels, ABC allows modeling at different degrees of details, facilitating (technical, syntactical and semantic) interoperability, autonomicity and distributed intelligence, and validating multiple design choices, before their actual deployment. In addition, agent-based methodologies can be extended and then reused for systematically driving the complete development process, also supporting integration with other paradigms (e.g., cloud computing, business process management) and reducing the probability of failure and time-to-market. However, before blindly adopting ABC in the IoT (as well as in any other development context), three main pragmatic aspects need to be considered, otherwise the overheads of dealing with agents could outweigh any benefits of an agent-based solution. First is related to relative immaturity of agent technology (born and raise mainly in the academia more than in the industry) and small number of available agent-based commercial platforms [15], which made no significantly progresses in the last decades [16], especially with respect to standardization and semantic interoperability (while, as discussed, such requirements are fundamental for the IoT). Second is related to the investment (in terms of both time and resources) needed for implementing agent-based IoT solutions, which are typically more costly than conventional centralized and service-oriented ones [47] (widely reused, for example, in the Web of Things [53]). Last one is related to old, but still common, misapplications and misconceptions, such as:

- *everything can be profitably agentified*: agents are intrinsically autonomous multi-thread problem solvers, thus an agent-based solution may be inappropriate for systems requiring only a single thread of control [15] (e.g., simple IoT monitoring applications) or unsustainable in the case of constrained IoT resources and devices [16] (unable, for example, to implement mechanisms for automatically handling conflicts among policies, synchronizing accesses to shared resources, developing distributed intelligence);
- *agents are a universal solution*: not all classes of IoT applications are suitable for agent-based techniques (for example, just 30% of control tasks and 60% of diagnostic tasks in the industrial scenario [47]).

Ultimately, as highlighted across this survey, we want to remark that the adoption of ABC paradigm needs to be carefully assessed but, as proved by the several contributions

presented in this work, it represents to date the most suitable choices for effectively developing the majority of advanced (current and future) IoT systems.

Acknowledgments. This work has been carried out under the framework of INTER-IoT, Research and Innovation action - Horizon 2020 European Project, Grant Agreement #687283, financed by the European Union. It was supported in part by PAS-RAS bilateral project "Semantic foundation of the Internet of Things", as well as a collaboration agreement between University of Novi Sad, University of Craiova, SRIPAS and Warsaw University of Technology.

References

1. Atzori, L., Iera, A., Morabito, G., 2010. The internet of things: A survey. Computer networks 54, 2787-2805.
2. Mattern, F., Floerkemeier, C., 2010. From the Internet of Computers to the Internet of Things, in: From Active Data Management to Event-Based Systems and More. Springer, pp. 242-259.
3. Patel, P., Cassou, D., 2015. Enabling high-level application development for the internet of things. Journal of Systems and Software 103, 62-84.
4. Luck, M., McBurney, P., Preist, C., 2004. A manifesto for agent technology: Towards next generation computing. Autonomous Agents and Multi-Agent Systems, 203-252.
5. Fortino, G., Guerrieri, A., Russo, W., Savaglio, C., 2014. Integration of agent-based and Cloud Computing for the smart objects-oriented IoT, in: Computer Supported Cooperative Work in Design (CSCWD), Proc. of the 2014 IEEE 18th Intl. Conf. on. IEEE, pp. 493-498.
6. Savaglio, C., Fortino, G., Zhou, M., 2016. Towards interoperable, cognitive and autonomic IoT systems: An agent-based approach, in: Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on. IEEE, pp. 58-63.
7. Jennings, N.R., 1999. Agent-based computing: Promise and perils.
8. Fortino, G., Rovella, A., Russo, W., Savaglio, C., 2016. Towards Cyberphysical Digital Libraries: Integrating IoT Smart Objects into Digital Libraries, in: Management of Cyber Physical Objects in the Future Internet of Things. Springer, pp. 135-156.
9. Fortino, G., Guerrieri, A., Russo, W., Savaglio, C., 2015. Towards a development methodology for smart object-oriented IoT systems: A metamodel approach, in: Systems, Man, and Cybernetics, 2015 IEEE Intl. Conf. on. IEEE, pp. 1297-1302.
10. Ricci, A., Santi, A., 2011. Agent-oriented computing: Agents as a paradigm for computer programming and software development, in: Proc. of the 3rd Intl Conf. on Future Computational Tech. and Applications. Wilmington: Xpert Publishing Services. Citeseer, pp. 42-51.
11. Poslad, S., 2007. Specifying protocols for multi-agent systems interaction. ACM Transactions on Autonomous and Adaptive Systems (TAAS) 2, 15.
12. Fipa, A., 2002. Fipa acl message structure specification. Foundation for Intelligent Physical Agents, http://www. fipa. org/specs/fipa00061/SC00061G. html (30.6. 2004).
13. Macal, C.M., North, M.J., 2005. Tutorial on agent-based modeling and simulation, in: Simulation Conference, 2005 Proc. of the Winter. IEEE, p. 14-pp.
14. Bergenti, F., Gleizes, M.-P., Zambonelli, F., 2006. Methodologies and software engineering for agent systems: the agent-oriented software engineering handbook. Springer Science & Business Media.
15. Wooldridge, M.J., Jennings, N.R., 1999. Software engineering with agents: Pitfalls and pratfalls. IEEE Internet Computing 3, 2027.

16. Nwana, H.S., Ndumu, D.T., 1999. A perspective on software agents research. The Knowledge Engineering Review 14, 125142.
17. Manzalini, A., Zambonelli, F., 2006. Towards autonomic and situation-aware communication services: the cascadas vision, in: Distributed Intelligent Systems: Collective Intelligence and Its Applications, 2006. DIS 2006. IEEE Workshop on. IEEE, pp. 383-388.
18. Spanoudakis, N., Moraitis, P., 2015. Engineering ambient intelligence systems using agent technology. IEEE Intelligent Systems 30, 60-67.
19. Katasonov, A., Kaykova, O., Khriyenko, O., Nikitin, S., Terziyan, V.Y., 2008. Smart Semantic Middleware for the Internet of Things. ICINCO-ICSO 8, 169-178.
20. Ruta, M., Scioscia, F., Loseto, G., Di Sciascio, E., 2014. Semantic-based resource discovery and orchestration in home and building automation: A multi-agent approach. IEEE Trans. on Industrial Informatics 10, 730-741.
21. Vlacheas, P., Giaffreda, R., Stavroulaki, V., Kelaidonis, D., Foteinos, V., Poulios, G., Demestichas, P., Somov, A., Biswas, A.R., Moessner, K., 2013. Enabling smart cities through a cognitive management framework for the internet of things. IEEE Communications Magazine 51, 102-111.
22. Fortino, G., Guerrieri, A., Russo, W., 2012. Agent-oriented smart objects development, in: Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th Intl. Conf. on. IEEE, pp. 907-912.
23. Cicirelli, F., Guerrieri, A., Spezzano, G., Vinci, A., Briante, O., Ruggeri, G., 2016. iSapiens: A platform for social and pervasive smart environments, in: Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on. IEEE, pp. 365-370.
24. Kato, T., Chiba, R., Takahashi, H., Kinoshita, T., 2015. Agent-Oriented Cooperation of IoT Devices Towards Advanced Logistics, in: Computer Software and Applications Conference, 2015 IEEE 39th Annual. IEEE, pp. 223-227.
25. Zhang, X., Adhikari, R., Pipattanasomporn, M., Kuzlu, M., Bradley, S.R., 2016. Deploying IoT devices to make buildings smart: Performance evaluation and deployment experience, in: Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on. IEEE, pp. 530-535.
26. Terziyan, V., Kaykova, O., Zhovtobryukh, D., 2010. Ubiroad: Semantic middleware for context-aware smart road environments, in: Internet and Web Applications and Services (Iciw), 2010 Fifth Intl. Conf. on. IEEE, pp. 295-302.
27. Mzahm, A.M., Ahmad, M.S., Tang, A.Y., 2013. Agents of Things (AoT): An intelligent operational concept of the Internet of Things (IoT), in: Intelligent Systems Design and Applications (ISDA), 2013 13th Intl. Conf. on. IEEE, pp. 159-164.
28. Ayala, I., Amor, M., Fuentes, L., 2015. The Sol agent platform: Enabling group communication and interoperability of self-configuring agents in the Internet of Things. Journal of Ambient Intelligence and Smart Environments 7, 243-269.
29. Leppänen, T., Riekki, J., Liu, M., Harjula, E., Ojala, T., 2014. Mobile agents-based smart objects for the IoT, in: Internet of Things Based on Smart Objects. Springer, pp. 29-48.
30. Pujolle, G., 2006. An autonomic-oriented architecture for the internet of things, in: Modern Computing, 2006. IEEE John Vincent Atanasoff 2006 Int.l Symp. on. IEEE, pp. 163-168.
31. Manate, B., Munteanu, V.I., Fortis, T.-F., 2013. Towards a scalable multi-agent architecture for managing iot data, in: P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth Intl. Conf. on. IEEE, pp. 270-275.
32. Wu, Q., Ding, G., Xu, Y., Feng, S., Du, Z., Wang, J., Long, K., 2014. Cognitive internet of things: a new paradigm beyond connection. IEEE Internet of Things Journal 1, 129-143.
33. VICINITY - Open virtual neighbourhood network to connect IoT infra-structures and smart objects, http://vicinity2020.eu/vicinity/
34. Kiljander, J., Delia, A., Morandi, F., Hyttinen, P., Takalo-Mattila, J., Ylisaukko-Oja, A., Soininen, J.-P., Cinotti, T.S., 2014. Semantic interoperability architecture for pervasive computing and internet of things. IEEE access 2, 856-873.

35. Cheng, B., Longo, S., Cirillo, F., Bauer, M., Kovacs, E., 2015. Building a big data platform for smart cities: Experience and lessons from santander, in: Big Data (BigData Congress), 2015 IEEE Intl. Congr. on. IEEE, pp. 592-599.
36. Ganzha, M., Paprzycki, M., Pawlowski, W., Szmeja, P., Wasielewska, K., 2017. Semantic interoperability in the Internet of Things: An overview from the INTER-IoT perspective. Journal of Network and Computer Applications 81, 111-124.
37. Savaglio, C., Fortino, G., 2015. Autonomic and Cognitive Architectures for the Internet of Things, in: Intl. Conf. on Internet and Distributed Computing Systems. Springer, pp. 39-47.
38. Mitrović, D., Ivanović, M., Budimac, Z., Vidaković, M., 2014. Radigost: Interoperable web-based multi-agent platform. Journal of Systems and Software 90, 167178.
39. Xu, X., Bessis, N., Cao, J., 2013. An Autonomic Agent Trust Model for IoT systems. Procedia Computer Science 21, 107113.
40. Karnouskos, S., De Holanda, T.N., 2009. Simulation of a smart grid city with software agents, in: Computer Modeling and Simulation, 2009. EMS09. Third UKSim European Symposium on. IEEE, pp. 424-429.
41. D'Angelo, G., Ferretti, S., Ghini, V., 2017. Multi-level simulation of Internet of Things on smart territories. Simulation Modelling Practice and Theory 73, 3-21.
42. Fortino, G., Russo, W., Savaglio, C., 2016. Simulation of Agent-oriented Internet of Things Systems, in: Proc. 17th Workshop" From Objects to Agents. pp. 8-13.
43. Fortino, G., Russo, W., Savaglio, C., 2016. Agent-oriented modeling and simulation of IoT networks, in: Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on. IEEE, pp. 1449-1452.
44. Zambonelli, F., 2016. Towards a General Software Engineering Methodology for the Internet of Things. arXiv preprint arXiv:1601.05569.
45. Kasnesis, P., Toumanidis, L., Kogias, D., Patrikakis, C.Z., Venieris, I.S., 2016. ASSIST: An agent-based SIoT simulator, in: Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on. IEEE, pp. 353358.
46. Manate, B., Fortis, F., Moore, P., 2014. Applying the Prometheus methodology for an Internet of Things architecture, in: Proc. of the 2014 IEEE/ACM 7th Intl. Conf. on Utility and Cloud Computing. IEEE Computer Society, pp. 435-442.
47. Marik, V., McFarlane, D., 2005. Industrial adoption of agent-based technologies. IEEE Intelligent Systems 20, 2735.
48. Razzaque, M.A., Milojevic-Jevric, M., Palade, A., Clarke, S., 2016. Middleware for internet of things: a survey. IEEE Internet of Things Journal 3, 70-95.
49. Fortino, G., Guerrieri, A., Russo, W., Savaglio, C., 2014. Middlewares for smart objects and smart environments: overview and comparison, in: Internet of Things Based on Smart Objects. Springer, pp. 1-27.
50. Kubler, O., S.; Framling, K.; Zaslavsky, A.; Doukas, C.; Olivares, E.; Fortino, G.; Palau, C. E.; Soursos, S.; Podnar Åarko, I.; Fang, Y.; Kro, S.; Heinz, C.; Grimm, C.; Broering, A.; Miti, J.; Olstedt, K.; Vermesan, O., 2016. Digitising the Industry: Internet of Things Connecting the Physical, Digital and Virtual Worlds. River Publishers, chapter 9, pp. 431-448, Vol. 49.
51. Morris, A., Giorgini, P., and Abdel-Naby, S., 2009. Simulating BDI-Based Wireless Sensor Networks. 2009 IEEE/WIC/ACM International Joint Conf. on Web Intelligence and Intelligent Agent Technology, volume 2, IEEE, pp. 78-81.
52. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J., 2004. Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems 8, 203236.
53. Guinard, D., Trifa, V., Pham, T., Liechti, O., 2009. Towards physical mashups in the web of things, in: Networked Sensing Systems (INSS), 2009 Sixth International Conference on. IEEE, pp. 14.