Simulation of Agent-oriented Internet of Things Systems

Giancarlo Fortino, Wilma Russo, and Claudio Savaglio
DIMES - Department of Informatics, Modeling, Electronics and Systems
University of Calabria
Via P. Bucci, cubo 41C, 87036 Rende (CS), Italy
{g.fortino, w.russo}@unical.it, csavaglio@dimes.unical.it

***Abstract*—The proliferation of everyday smart devices able to sense, process, communicate and/or actuate, is changing the way we interact with the world around us. These novel cyber-physical smart devices, or simply Smart Objects (SOs), are the fundamental building blocks of the Internet of Things (IoT), a global and highly dynamic ecosystem in which heterogeneous typologies of device networks seamlessly interoperate. Although the IoT component technologies and enabling computing paradigms are not totally new, the development and analysis of an IoT system is still a complex process. In this paper the ACOSO (Agent-based COoperative Smart Object) middleware and the Omnet++ platform have been used as means for the SO-based IoT systems development/management and simulation. Indeed, on one hand ACOSO provides effective instruments and a simple programming model to realize both cyber-physical SOs and IoT systems. On the other hand, leveraging on the parallelism between SOs/agents and the Omnet++ network nodes, simulations of agent-oriented IoT systems in different scale scenarios have been defined and conducted.**

Keywords—Internet of Things; Smart Objects; ACOSO; Agent-based computing; Network simulation.

I. INTRODUCTION

The Internet of Things (IoT) is being widely considered as the next revolution towards the digitalization of our society and economy, overturning the current production of goods and services. Only in the European Union, the IoT market value is expected to exceed one trillion euros in 2020, when it is foreseen that almost 26 billion of IoT devices will daily impact our life [1]. People, things and places will participate in the Internet, being globally identified, interconnected, discovered and queried. Everything will automatically but seamlessly interact, even without a steady human orchestration, thus providing novel cyber-physical services to both humans and machines. Although the IoT vision (of an horizontal and interconnected landscape) is unique and well-established, over the years three perspectives raised [2] that respectively emphasized the importance of the IoT devices, communication networking and semantic technologies. Although it is widely recognized that a variety of technologies and research areas contributes to the IoT, the "Things oriented" perspective is gaining more and more attention. As matter of fact, such "things" or smart objects (SOs) have been defined as fundamental IoT blocks [3] since they concretely and daily realize the bridging between the real and the virtual world. SOs, beside their specific purpose (e.g. refrigerating foods in the case of a smart fridge), are indeed able to sense the surrounding physical environment, elaborate the perceived data, share them (with human users or with other SOs) through adequate communication interfaces and, if necessary, to take tangible actions. The physical proximity or the similitude of purposes among multiple SOs enable the constitution of IoT systems, in which functional, technological and application-specific heterogeneities should not prevent SOs to interoperate with each other. In order to cope with all such issues, the IoT has drawn several concepts from multiple paradigms (e.g. cloud computing, web-services, etc.), including from the Agent-Based Computing (ABC) paradigm [4]. In the past years, research and industrial experiences in a wide range of application domains (e.g. logistics, economics, social science, automation science) have already proved the advantages deriving from the use of the ABC in developing complex distributed systems under the form of MASs [5]. In the case of IoT, that can be itself considered as a loosely coupled, decentralized system of cooperating SOs, the ABC is even more suitable to support the development of the single SO ("in the small") and of the overall IoT system ("in the large"). However, the IoT systems development process based on ABC is still in its infancy. Within such development process, the simulation activity plays a crucial role: indeed, it allows the understanding of system/network performance and dynamics before the actual SO-based system implementation and deployment. In such case too, however, further efforts need to be made.

In this paper we propose the simulation of agent-oriented IoT systems in small-medium-large scale scenarios through the Omnet++ simulation platform [6], with a specific attention to the inter-SO communications phase. By following such approach, low-level aspects (wireless coverage issues, physical environment and obstacles modeling, protocols implementation details, etc.) are managed by Omnet++ while the ACOSO middleware [7, 8] provides an effective agent-oriented SO programming and design model. Indeed, ACOSO is specifically conceived for the SO development, management and deployment in any application context which requires proactivity and reactivity with respect to the surrounding environment and to other SOs. The rest of the paper is organized as follows. In Section II, the background of the SO-based IoT is provided, together with a brief related work on the available agent-oriented IoT contributions. In Section III, the

This work has been partially carried out under the framework of INTER-IoT, Research and Innovation action - Horizon 2020 European Project, Grant Agreement #687283, financed by the European Union.

ACOSO middleware is briefly summarized, focusing on its multi-layered and agent-oriented architecture. Simulations of agent-oriented IoT systems characterized by different scales and configurations are reported in Section IV. Conclusion and future work are finally delineated.

II. Background and Related Work

A. Smart Object-based IoT

The advancements on integrated circuitry, microelectro-mechanical systems (MEMS), embedded technologies, and wireless communications enabled the evolution of conventional everyday things in enhanced entities commonly defined Smart Objects (SOs) [3]. Differently from passive RFiD systems and conventional Wireless Sensor Networks (WSNs), an SO is able to provide identification, sensing/actuation but also to understand and react to its environment [9], performing object-to-object communications, ad-hoc networking and complex goal-oriented decision-making. SOs with limited computational resources (e.g. RAM, CPU) may be usefully supported by the Cloud computing [10], which enables devices virtualization and dynamic data processing (e.g. data integration/fusion). More powerful SOs, instead, may be designed by following the principles of autonomic computing and of the cognitive networks, in order to become even more autonomous, proactive, context-aware and intelligent [11]. In both cases, SOs are suitable to replace the human operators in handling the seamless data flow between different networks typologies, like BAN (Body Area Networks), LAN (Local Area Networks, e.g. Smart Home), MAN (Metropolitan Area Networks, e.g. Smart Hospital) and WAN (Wide Area Networks, e.g. Smart City). On the other hand, a steady human orchestration of such a huge amount of devices and device-generated data is not feasible in the IoT context. The synergic cooperation of multiple SOs may in fact generate complex and outstanding cyber-physical services for both humans and machines, but only if the SOs are adequately designed and implemented. In fact, the SOs are usually functional and technological heterogeneous with each other, following different communication protocols and data formats standards on the basis of their application domains. Such issues are currently leading to the spread of several IoT silos that are unable to interoperate [12], preventing the fruition of the benefits of a fully-realized global IoT. So, proper development methodologies, modeling paradigms, software abstractions and interaction patterns need to be adopted by design [13] in order to overcome SOs heterogeneities and to make SOs completely interoperable.

B. Agent-oriented IoT

The IoT ecosystem development process includes multiple requirements, both at system (e.g. scalability, robustness, standards compliance, discovery) and at thing level (e.g. interoperability, virtualization, embedded intelligence). The ABC offers the necessary solutions to satisfactorily address such requirements by running agents in IoT nodes and hence by treating the IoT ecosystem as a MAS. The idea of tightly coupling each SO with (at least) one agent [14] has multiple benefits since the agent(s) allows mitigating the SO host hardware/software deficiencies or limitations. In fact, agents are able to encapsulate complex functionalities abstracting them from the underlying implementation details, communicate over different access technologies simultaneously, interact with pro-activeness, autonomy and sociability. Consequently, agents running in different cooperating SOs constitute a decentralized MAS, maximizing interoperability among heterogeneous sub-systems and distributed resources, facilitating the system modeling and development, increasing scalability and robustness but, at the same time, reducing the design time as well as the time-to-market. These motivations have driven the design of several agent-based IoT architectures [11, 15-21] that exploit the twofold ABC role of:

(i) *Modeling paradigm*, because most of the SO main features may be described through agent-related concepts. For example, SO functionalities may be expressed in terms of goals, SO working plan in terms of behaviors, SO augmentation devices (e.g. sensors and actuators) in terms of dynamically bindable agent resources, etc. In this direction, [15] and [16] propose coarse-grained agent-oriented SO models, characterized by a high-degree of abstraction to support the preliminary development phase of SO analysis.

(ii) *Programming paradigm*, by exploiting the agent as a virtual networked alias of the real object [17, 21]. The virtualization process allows the integration of the SOs in the cloud or in the SOA/REST world [19], enabling even constrained SOs to provide complex cyber-physical services. In such directions, the virtualization allows also the federation of semantically interoperable SOs, enabling the mashup of their offered services in accordance with both the application and user requirements [18].

A particular component that plays a crucial role within most of the distributed architectures is the middleware. In the context of the IoT systems, different agent-based middleware have been developed so far [9, 20] since the ABC provides powerful mechanisms to realize efficient coordination structures, SO discovery, resources handling and knowledge management. As matter of fact, the exploitation of well-established agent communication standards and interfaces (e.g. IEEE FIPA [22]) contributes in hiding the SO heterogeneities both at physical and at communication layers. Moreover the ABC allows the development of both semi/centralized (following the IEEE FIPA model that foresees the Directory Facilitator for mapping agents and their services) and distributed (by following a P2P approach) service discovery. Such features have particular importance since the IoT is a dynamic scenario in which SOs seamlessly appear, disappear, as well as extemporary interact with each other. The application of the ABC at middleware layer is also suitable to integrate agents with semantic technologies (e.g. ontology), facilitating the data and the context management as well as the implementation of security mechanisms. Doing so, agents provide intelligence, context-awareness, robustness and flexibility to single SOs as well as to the whole IoT system.

III. ACOSO (Agent-based COoperating Smart Object)

ACOSO (Agent-based COoperating Smart Objects) [7, 8] is a middleware providing a (in-the-small and in-the-large) SO programming model through an agent-based approach. ACOSO presents an event-driven and multi-layered architecture that allows the SOs to react to external stimulus,

fulfill specific goals, execute inference rules, and use local/remote knowledge bases. Following a bottom-up approach, the ACOSO platform presents the following layers:

i. *WSAN management layer*, which programs and manages the network of sensors and/or actuators embedded in a SO. Such layer allows the management of WSANs (Wireless Sensor and Actuator Networks) through the BMF (Building Management Framework) [23] and of body sensor networks through the SPINE (Signal In-Node Processing Environment) [24].

ii. *Agent-based middleware layer*, relying on the JADE platform, that provides an effective agent-oriented management/communication infrastructure. In particular, JADE-based SOs are managed by the AMS (Agent Management System), communicate through the ACL-based message transport system and use an extended version of the DF (Directory Facilitator) to look up SOs and other agents. JADE provides also a coordination model implementing both the message passing (MessagingService) and the publish/subscribe (TopicManagementService) communication paradigms through a ServiceManager. The original JADE DF, indeed, has been purposely modified/extended in ACOSO to support a more situated and dynamic SO registration, indexing and discovery on the basis of its specific functional (e.g. provided services) and/or non-functional (e.g. location, dimension, identity) features. Since the JADE platform may run both on Java-enabled and Android devices (by means of LEAP, a JADE extension), this layer can concretely implement the high-level SO layer atop PC, smartphones, tablets, etc.

iii. *High-level SO layer*, which comprises a set of subsystems describing the SO internal architecture. In detail, each SO goal is encapsulated in state-based tasks, which are driven by events and managed by the Task Management Subsystem. The Communication Management Subsystem provides a common interface for SOs communications: the Communication Manager Message Handler translates incoming messages into internal events that are managed by the EventDispatcher. The Device Management Subsystem manages the SO sensor/actuator devices by means of specific DeviceAdapters. The KB Management Subsystem manages the object knowledge base. In such subsystems an important role is played by the adapters that represent pluggable software components allowing SOs to interoperate with external entities or systems. For example, within the Device Management Subsystem, two DeviceAdapters are currently defined to interact with the WSAN management layer: the BMFAdapter, which allows managing WSANs through the BMF [25], and the SPINEAdapter, which allows managing BSNs through SPINE [26]–[28]. Within the Communication Management Subsystem, instead, the TCPAdapter and UDPAdapter manage SO communication with external networked entities based on TCP and UDP, respectively. The aforementioned agent-oriented subsystems that compose the High-level SO layer are platform neutral but, at the Agent-based middleware layer, the Tasks, the EventDispatcher and the Communication Manager Message Handler have been implemented as JADE Behaviors (so their execution is based on the mechanisms provided by the basic JADE behavioral execution model), while the SO messages are defined as ACL messages.

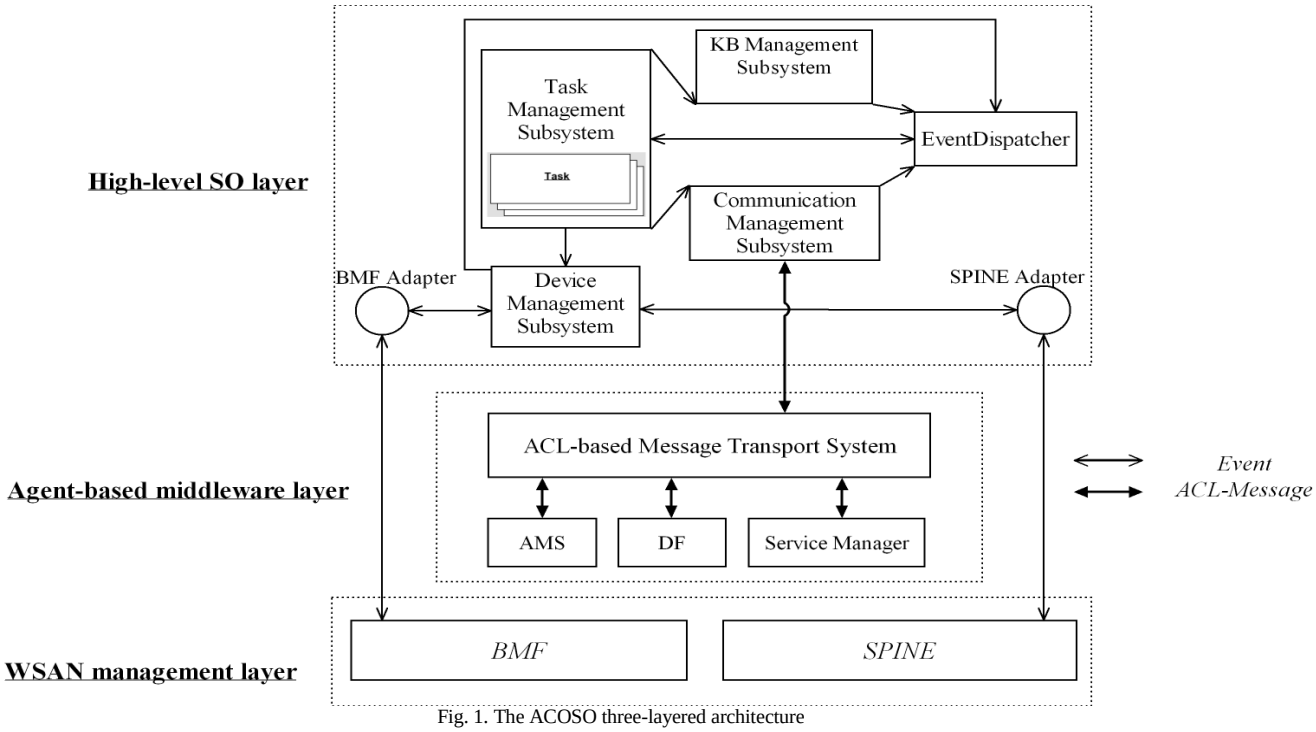

Fig. 1. The ACOSO three-layered architecture

IV. Simulations

The simulation of IoT systems allows the validation of models, protocols and algorithms before the actual SO deployment phase. Due to such reasons, it is an important but, at the same time, challenging task. In IoT systems of different scales the number of the SOs may vary (from body sensor networks with less than dozens of SOs, to Smart City with much more than thousands of devices), with a different degree of density, as well as the SO services require different communication paradigms. Just the SO interactions and the consequent service provision/fruition may be influenced by factors unrelated to the applications but specifically associated to the networking (e.g. traffic congestion, wireless signal attenuation and coverage, etc.). In this paper, we focused on the communication among SOs (previously modeled with the ACOSO approach) by simulating IoT networks through Omnet++[6]. As matter of facts, the parallelism between SOs/agents and Omnet++ network nodes is straightforward. In fact, each network node can be considered as an autonomous SO/agent whose behaviors and tasks can be implemented at the application layer. All the other tasks related to transport-network-link protocol implementations, wireless connectivity issues, physical environment modeling can be instead carried out by Omnet++. In the following, the results of IoT systems simulations are shown, with a particular attention to the inter-SO communication. These simulations aim at investigating possible issues or unpredicted situations, and at validating IoT systems design choices and parameters. Application-neutral scenarios and SOs exchanging empty messages without any additional application logic have been considered, thus providing more generality to the obtainable simulation results.

A. Communication settings

Simulations have inspected IoT networks in the **SO Discovery phase (SOD)** and in the **Information Exchange phase (IE)** by exploiting in both cases reliable (TCP) and unreliable (UDP) transport protocols. Metrics, parameters and patterns that have been tested in the simulations are listed as follows:

- **Metrics**: in the SOD phase the discovery time (DT) and the request delivery ratio (RDR) have been measured; in the IE phase the round trip time (RTT) and the message delivery ratio (MDR).
- **Parameters**: in the SOD phase all the nodes, with a different request generation rate (RGR), contact a specific one that holds the registry of the current active SOs and of their provided services. In the IE phase the round trip time (RTT) and the message delivery ratio (MDR) have been measured when nodes adopt different message generation rates (MGR). Both in SOD and in IE phases, different request/data generation models (RGM and DGM, respectively) are used: a Deterministic one (1 pk/s and 10 pk/s) and a stochastic Normal one (with 0.5 mean and 0.2 variance).
- **Patterns**: in the SOD phase SOs communicate according to the Client/Server (C/S) paradigm; in the IE phase, SOs exchange simple messages by following either a C/S or a Peer-to-Peer (P2P) paradigm.

B. Simulation Scenarios

Performance metrics presented in Section IV-A have been evaluated in the context of small-, medium-, large-scale IoT networks with different SOs density. In particular, since network congestion may increase depending of the SOs population, the performance metrics have been analyzed when the number of the SOs (#SOs) increases. Small-scale networks have been considered limited to 100 nodes, medium-scale networks to 500 nodes and large-scale networks to 1000. Moreover, it has been analyzed how the SO distribution in a different number of subnetworks (#subnetworks) impacts the performance metrics. It has been assumed that: (i) small-scale networks are constituted by a single network; (ii) medium-scale networks consist of two or more adjacent subnetworks (which are deployed in the same area so that their coverage overlaps); (iii) large-scale networks include multiple but distinct subnetworks (their coverage does not overlap).

C. Results

With respect to the small-scale network, Fig. 2a shows that in the SOD phase the increase of the SO population adversely affects the DT, as well as a high RGR and the choice of a reliable transmission protocol.

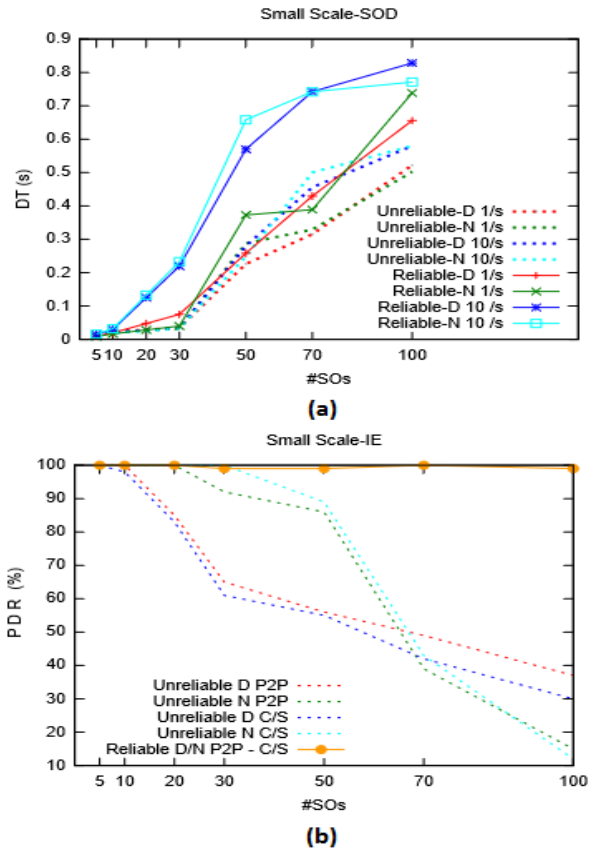

Fig. 2. Small scale networks: DT in SOD phase (a) when the #SOs changes; PDR in IE phase (b) when the #SOs changes.

Such phenomena are particularly remarked when the SOs exceed the 30 units while there are no consistent differences between deterministic (D) or normal (N) data sources. In IE phase, the increase of the SOs reduces the PDR only in the case of unreliable protocol as shown in Fig 2b.

With respect to the medium-scale networks, Fig 3a highlights that in the SOD phase if SOs are equally spread on multiple subnetworks (we consider the case of five subnetworks deployed in a squared grid with side of 2500 meters), the increase of the SOs number causes the DT increase, while unreliable protocols, lower MGR or normal data sources provide smaller DT values. Such trends are similar to the ones related to the small-scale case. Fig 3b shows that if the SOs are distributed on the same area, the RTT decreases because the traffic is balanced on more subnetworks. In such a case, the P2P paradigm outperforms the C/S.

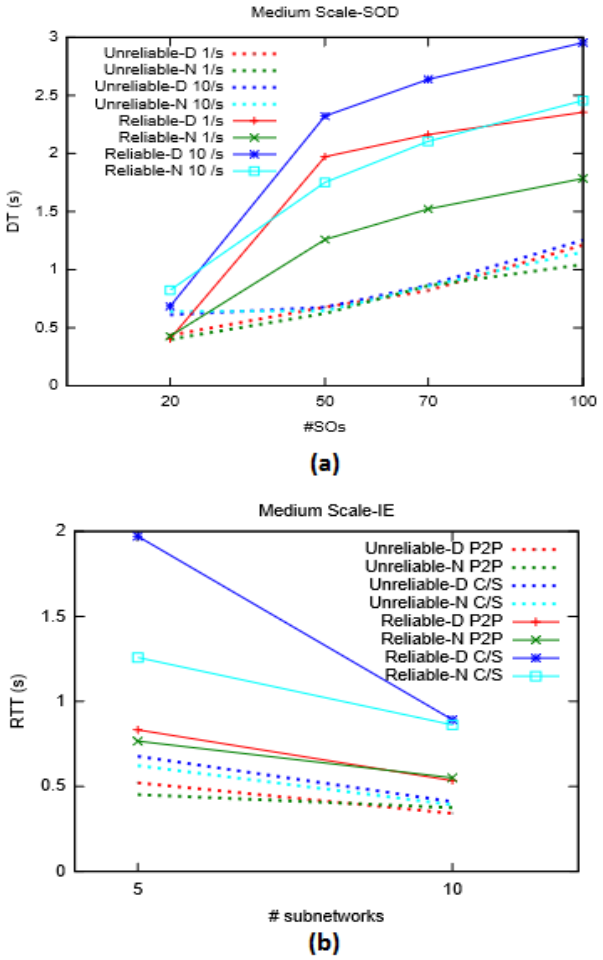

Fig. 3. Medium-scale networks: DT in SOD phase (a) when the #SOs changes (a); RTT in IE phase (b) when the #subnetworks changes.

With respect to large-scale networks, a different number of non-overlapping subnetworks (5, 10, and 20) has been considered, each one with the same number (50) of SOs. As Fig. 4a highlights, since the subnetworks have no overlap, the absence of mutual interferences makes the DT quite stable (the subnetworks performance in the SOD phase is similar to the single network case of the small-scale scenario). Again, the reliable protocol as well as the 10/s MGR implies greater DT while there are no substantial differences between D or N data sources. These considerations hold for both the SOD and the IE phases, as Fig. 4b shows. In particular, DT values of large-scale multiple subnetworks scenario are comparable to the ones of the small-scale single network scenario.

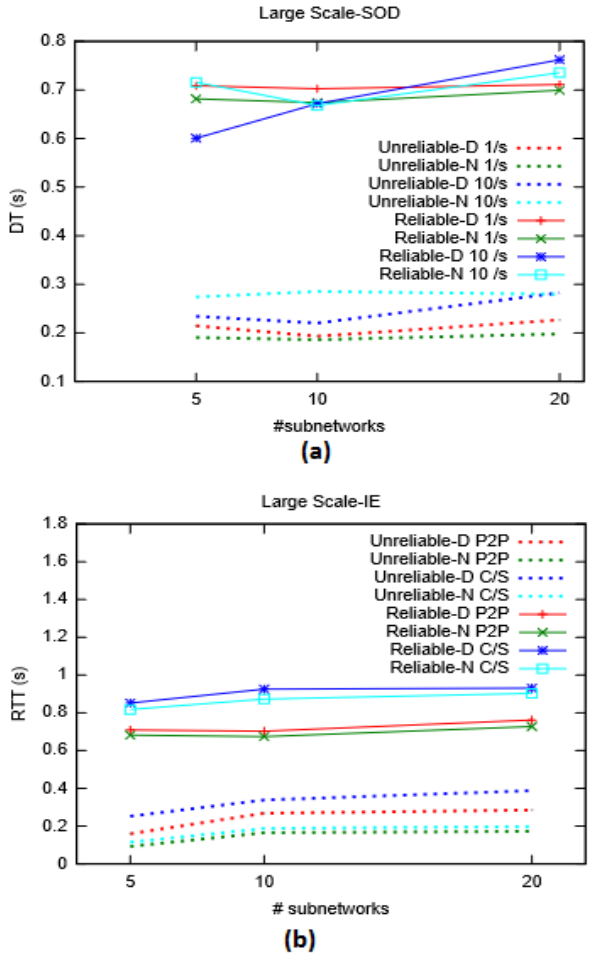

Fig. 4. Large-scale networks: DT in SOD phase (a) when the #subnet changes (a); RTT in IE phase (b) when the #subnetworks changes.

V. Conclusion

The agent-based programming paradigm definitively represents a viable approach to support the development of the distributed and heterogeneous elements of a SO-oriented IoT. Indeed, the agent abstraction provides an efficient and powerful way to describe the SOs, that are self-contained, autonomous, social, adaptive, and goal-directed building blocks of IoT systems. At the same time, such IoT systems may be treated as MASs, since they are dynamic, self-organized and situated ecosystems. To facilitate the SOs development process and speed up the IoT elements prototyping phase, middleware solutions have been proposed since they provide useful general and specific abstractions at different levels of granularity. The agent-oriented ACOSO middleware represents an effective framework for the developing of SOs able to perform

distributed computation, knowledge management and flexible interaction with sensors and actuators devices. Beside the SOs development process, the simulation of the under-development IoT system is an equally important activity. Through the Omnet++ platform, a set of simulations in different scale scenarios has been performed and the related results presented, with the focus on the communications between SOs. Regardless of the considered small-medium-large scale, simulation results highlight that the increase of SOs number contributes to the network traffic, so causing lower performance, especially if reliable protocols are adopted. This implies that such kind of protocols should be adopted only if the full reliability is a mandatory requirement. Performance is also adversely affected by an unbalanced traffic load, which may be due to the adoption of centralized communication paradigms like C/S or by deterministic data sources. Both in the medium- and in the large-scale cases, trends related to the increase of SOs number and of their density, or about protocols reliability, MGR and stochastic/deterministic data sources, hold. However, with respect to medium-scale networks, the presence of multiple overlapping subnetworks on the same area produces interferences, so reducing the performance. If there are no overlaps among the subnetworks as in the large-scale case, instead, the communications are scalable and the provided performance is improved. Future work will be focused on the design of a methodology that systematically supports the complete development of SO-based IoT systems [13]: such methodology, by integrating ACOSO and Omnet++ with other frameworks like ELDA [29-31] and by maintaining an agent-oriented approach, will aim to effectively model, prototype and validate new generation of SO-based cyber-physical systems within the IoT context.

REFERENCES

[1] S. Aguzzi, et al., "Definition of a Research and Innovation Policy Leveraging Cloud Computing and IoT Combination," European Commission, Directorate-General of Communications Networks, Content & Technology, Belgium, Rep. SMART 2013/0037, 2014.

[2] D. Miorandi, D. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," Ad Hoc Networks, vol. 10, no. 7, 2012, pp. 1497-1516.

[3] G. Kortuem, F. Kawsar, D. Fitton, V. Sundramoorthy, "Smart objects as building blocks for the internet of things,". Internet Computing, IEEE, vol. 14, no. 1, 2010, pp. 44-51.

[4] M. J. Wooldridge, "Agent technology: foundations, applications, and markets," Springer Science & Business Media, 1998.

[5] W. Van der Hoek, and M. J. Wooldridge, "Multi-agent systems," Handbook of Knowledge Representation, 2008, pp. 887-928.

[6] A. Varga, "The OMNeT++ discrete event simulation system," Proceedings of the ESM'2001, vol. 9, no. S 185, 2001, p. 65.

[7] G. Fortino, A. Guerrieri, M. Lacopo, M. Lucia, and W. Russo, "An agent-based middleware for cooperating smart objects," Highlights on Practical Applications of Agents and Multi-Agent Systems, Springer Berlin Heidelberg, 2013, pp. 387-398.

[8] G. Fortino, A. Guerrieri, and W. Russo, "Agent-oriented smart objects development," Computer Supported Cooperative Work in Design, 2012 IEEE 16th International Conference on. IEEE, 2012, pp. 907-912.

[9] G. Fortino, A. Guerrieri, W. Russo, and C. Savaglio, "Middlewares for smart objects and smart environments: Overview and comparison," Internet of Things Based on Smart Objects. Springer International Publishing, 2014, pp. 1-27.

[10] G. Fortino, A. Guerrieri, W. Russo, and C. Savaglio, "Integration of agent-based and cloud computing for the smart objects-oriented IoT," Computer Supported Cooperative Work in Design, Proceedings of the 2014 IEEE 18th International Conference on, 2014, pp. 493-498.

[11] C. Savaglio, and G. Fortino, "Autonomic and Cognitive Architectures for the Internet of Things," Internet and Distributed Computing Systems, Springer International Publishing, 2015, pp. 39-47.

[12] P. Desai, A. S. Pratikkumar, and A. Pramod, "Semantic gateway as a service architecture for iot interoperability," Mobile Services (MS), 2015 IEEE International Conference on, 2015, pp. 313-319.

[13] G. Fortino, A. Guerrieri, W. Russo, and C. Savaglio, "Towards a Development Methodology for Smart Object-Oriented IoT Systems: A Metamodel Approach," Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on, 2015, pp. 1297-1302.

[14] G. Fortino, "Agents Meet the IoT: Toward Ecosystems of Networked Smart Objects", IEEE Systems, Man & Cybernetics Magazine, April 2016. doi: 10.1109/MSMC.2016.2557483.

[15] T. Leppänen, J. Riekki, M. Liu, E. Harjula, and T. Ojala, "Mobile agents based smart objects for the internet of things," Internet of Things Based on Smart Objects, Springer International Publishing, 2014, pp. 29-48.

[16] A. Katasonov, et al. "Smart Semantic Middleware for the Internet of Things," ICINCO-ICSO, vol. 8, 2008, pp. 169-178.

[17] A.M. Mzahm, M.S. Ahmad, and A. YC Tang, "Agents of Things (AoT): An intelligent operational concept of the Internet of Things (IoT)," Intelligent Systems Design and Applications (ISDA), 2013 13th International Conference on. IEEE, 2013, pp. 159-164.

[18] D. Uckelmann, M. Harrison, and F. Michahelles, "An architectural approach towards the future internet of things," Springer Berlin Heidelberg, 2011.

[19] I. Ayala, M. Amor, and L. Fuentes, "An agent platform for self-configuring agents in the internet of things," Infrastructures and Tools for Multiagent Systems, 2012, p. 65.

[20] S. Bandyopadhyay, M. Sengupta, S. Maiti, and S. Dutta, "Role of middleware for internet of things: A study," International Journal of Computer Science and Engineering Survey, vol.2, no.3, pp. 94-105.

[21] P. Vlacheas, et al., "Enabling smart cities through a cognitive management framework for the internet of things," Communications Magazine, IEEE, vol. 51, no. 6, 2013, pp. 102–111.

[22] F. Bellifemine, A. Poggi, and G. Rimassa, "JADE–A FIPA-compliant agent framework," Proceedings of PAAM, vol. 99, 1999, pp. 97-108.

[23] G. Fortino, A. Guerrieri, G. M. O'Hare,and A. Ruzzelli, "A flexible building management framework based on wireless sensor and actuator networks," Journal of Network and Computer Applications, vol. 35, no. 6, 2012, pp. 1934-195.

[24] G. Fortino, R. Giannantonio, R. Gravina, P. Kuryloski, and R. Jafari, "Enabling effective programming and flexible management of efficient body sensor network applications," IEEE Transactions on Human-Machine Systems, vol. 43, no. 1, 2013, pp. 115-133.

[25] A. Guerrieri, G. Fortino, A. Ruzzelli, and G. O'Hare, "A WSN-based Building Management Framework to Support Energy-Saving Applications in Buildings," Hershey, PA, USA, IGI Global, 2011.

[26] G. Fortino, S. Galzarano, R. Gravina, and W. Li, "A framework for collaborative computing and multi-sensor data fusion in body sensor networks," Information Fusion, vol. 22, 2015, pp. 50-70.

[27] G. Fortino, A. Guerrieri, F. Bellifemine, and R. Giannantonio, "SPINE2: developing BSN applications on heterogeneous sensor nodes," IEEE Int. Symposium on Industrial Embedded Systems, 2009, pp. 128-131.

[28] F. Bellifemine, G. Fortino, A. Guerrieri, and R. Giannantonio, "Platform-independent development of collaborative Wireless Body Sensor Network applications: SPINE2", SMC 2009, pp. 3144-3150.

[29] G. Fortino, A. Garro, S. Mascillaro, and W. Russo, "Using event-driven lightweight DSC-based agents for MAS modelling," Int. Journal of Agent-Oriented Software Engineering, vol. 4, no. 2, 2010, pp. 113-140.

[30] G. Fortino, and W. Russo, "ELDAMeth: An agent-oriented methodology for simulation-based prototyping of distributed agent systems," Information and Software Technology, vol. 54.6, 2012, pp. 608-624.

[31] G. Fortino, A. Garro, and W. Russo, "An integrated approach for the development and validation of multi-agent systems," Int. Journal of Computer Systems Science & Engineering, vol. 20.4, pp. 259-271, 2005.