Proceedings of the Federated Conference on Computer Science and Information Systems pp. 1449–1452
DOI: 10.15439/2016F359
ACSIS, Vol. 8. ISSN 2300-5963

Agent-oriented Modeling and Simulation of IoT Networks

Giancarlo Fortino
DIMES - University of Calabria
Via P. Bucci, cubo 41C, 87036
Rende (CS), Italy
g.fortino@unical.it

Wilma Russo
DIMES - University of Calabria
Via P. Bucci, cubo 41C, 87036
Rende (CS), Italy
w.russo@unical.it

Claudio Savaglio
DIMES - University of Calabria
Via P. Bucci, cubo 41C, 87036
Rende (CS), Italy
csavaglio@dimes.unical.it.

Abstract—**Internet of Things (IoT) networks are being continually developed in several domains, however no systematic processes for their modeling and simulation exist so far. In this paper, an agent-oriented approach to IoT networks modeling is proposed by exploiting the ACOSO model. Then, agent-modelled IoT networks of different scales are simulated through the Omnet++ simulation platform, with the goal of analyzing issues and bottlenecks at communication level.**

I. Introduction

SMART Objects (SOs) constitute a new generation of enhanced everyday things (able to perceive the surrounding physical environment, elaborate and communicate the acquired information, and hence provide cyber-physical services) that are globally networked and mutually interacting, even without a steady human orchestration [1]. SOs are technologically and functionally heterogeneous, thus their clustering constitutes "Internet of Things" (IoT) networks [2] that look like loose collections of heterogeneous devices and sub-networks requiring distributed mechanisms of communication and management. Whereas defining methods to model, design and simulate IoT networks before their final implementation is an open challenge, no research efforts have currently been devoted to systematically address such issue.

As IoT networks share many substantial features/issues with multi-agent systems (MAS) [3] and SOs may be effectively modeled as agents, in this paper an agent-oriented approach to model IoT networks is presented. In particular, the ACOSO (Agent-based Cooperative Smart Object)-based SO model [4] has been exploited to describe, from the agent perspective, SO main features, relationships and interactions. Moreover, in order to analyze IoT networks at SO communication level, the Omnet++ [5] network simulation platform has been used to evaluate bottlenecks and issues in specifically defined scenarios. The rest of the paper is organized as follows: in Section II, the ACOSO-based SO model and other concrete examples of the Agent-oriented Modeling (AoM) applied to the SO-based IoT context are introduced and compared. In Section III, the designed simulation scenarios, the selected metrics, and the obtained results are presented and discussed. Finally, conclusions are drawn and future work is briefly delineated.

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.

II. Agent-oriented Modeling

Agent-based Computing paradigm has been successfully used over the years for the analysis, modeling and implementation of complex, cooperative and adaptive distributed systems [3]. The *agent* abstraction, indeed, is suitable to model and implement autonomous, intelligent and interacting entities, characterized by different behaviors and goals. Nevertheless is well-established that the AoM [6], differently to other modeling paradigms (e.g. object-oriented, service-oriented, etc.) is able to fully support proactiveness and situatedness, key features in SO-based IoT networks, to date few agent-oriented models are available in the literature in this context.

With reference to the agent-oriented SO modeling, [7] and [8] present coarse-grained models, characterized by a high degree of abstraction and therefore mostly suitable to support the SO analysis phase; the ACOSO-based SO model [4], instead, specializes the SO ecosystem in a deeper degree of detail, thus supporting also the SO design and implementation phases. In detail, in [7] the authors envision an IoT system architecture where each resource (e.g. computer, SO, human user) is represented by an agent and interconnected to the rest of the cyber-physical world by means of specific adapters. Each agent has a role (and not an identifier) that determines its own behaviors, tasks and communication paradigms. Both roles and behaviors are taken from two repositories which are assumed to be managed depending on the application scenario. In [8] the SO model comprises five elements: the execution environment (to run the actual agent task and manage its lifecycle), a repository (which contains both database and knowledge base), a set of physical components (such as sensors and actuators), the agent interface (to enable the intra-agent information exchanges among the aforementioned SO-model elements) and the object interface (for communication with other SOs and with the system).

The ACOSO-based SO model allows the modeling of high-level SO main features (basic information of an SO, its augmentation devices like sensors and actuators, its services, etc.) and it also extensively describes the functional components of the system, their relationships and interactions. Due to such reasons, the ACOSO-based SO model represents one of the cornerstones of ACOSO (Agent-based Cooperative Smart Object) [9], a middleware specifically conceived for the full management and development of agent-oriented cooperating SOs. In detail, the ACOSO-based SO model abstracts SOs in event-driven cooperating agents whose specific objectives are encapsulated in their behavior and modeled as *Tasks*. A Task is an event-driven and state-based component that can refer to the common operations required for the agent lifecycle management (SystemTask) or to specific-purpose operations defining the specific behaviors of the SO (UserDefinedTask). Indeed, SO services are mapped on UserDefinedTask. By means of different tasks the SO exploits different subsystems in order to react to external stimulus, to fulfill specific goals and to exploit inference rules on local/remote knowledge bases. In particular:

- The *DeviceManagementSystem*, through multiple DeviceAdapters, handles the SO augmentation devices that enables SO to interact with the physical world generating Device Events. In particular, the BMFAdapter [10] and the SPINEAdapter [11-13] enable the management of environmental and wearable sensor networks.
- The *CommunicationManagementSystem* provides communication services between agents and external entities. Different kinds of interactions (e.g. intra-agent FIPA-ACL based interactions or inter-entities UDP/TCP-based interactions) are enabled by means of different CommunicationAdapters. Both events generated inside (InternalEvent) or outside (ExternalEvent) the SO are handled by the CommunicationManagementSystem.
- The *KBManagementSystem* exploits local or remote knowledge bases to handle information pertaining the SO, its current status, its inference rules and other useful data that can be shared among the agent tasks.

TABLE I
AGENT-BASED SO MODELS COMPARISON

Agent-based SO Modelling	**[7]**	**[8]**	**ACOSO-based SO model [4]**
SO-Modeling Phase / *SO Characteristic*	Analysis	Analysis	Analysis & Design & Implementation
Augmentation Devices	Adapter	Physical Components	DeviceManagementSystem (DeviceAdapter)
Communication	Role	Int./Ext. Agent Interface	CommunicationManagementSystem (CommunicationAdapter)
Decision-Making	Role	Execution Environment	Set of Task (Agent-Behavior)
Service Provisioning	N/A	N/A	UserDefinedTask, Event
Knowledge	Role	Repository	KBManagementSystem, Behavior

A. Comparison

Table I shows a comparison of the three agent-oriented SO models introduced in this Section. First of all, it has been highlighted that they are suitable in different modeling phases. Indeed, [7] and [8] are mostly suitable to approach the SO analysis phase while the ACOSO-based model support also design and implementation phases. This implies that the three models describe the main SO characteristics (augmentation, communication, decision-making, service provisioning and knowledge) with a different degree of detail. In particular:

- *Augmentation Devices*: in SO models of [7] and [8] sensors and actuators are not explicitly modeled as they are considered minor SO components and their interactions are not elicited. In the ACOSO-based SO model augmentation devices are managed by the DeviceManagementSystem and handled by multiple DeviceAdapters.
- *Communication*: in [7] the SO role determines its communication paradigm and structure, while in [8] internal/external SO communication interfaces are defined. In the ACOSO-based SO model communication is managed through the CommunicationManagementSystem and a set of CommunicationAdapters.
- *Decision-Making*: in [7] an SO/agent autonomously behaves to achieve certain goals depending on its role in the domain (e.g. smart car, smart driver-support or smart road within the smart transportation domain), while in [8] task execution concerns the Execution Environment. Decision-making relies on the agent behaviors in ACOSO-based SO model, defined through tasks.
- *Service Provisioning*: in [7] and [8] the concept of SO cyber-physical service is not completely declined and is mostly reduced to a simple Web Service. In the ACOSO-based SO model services are mapped on UserDefinedTasks (enabled by specific events).
- *Knowledge Base*: in [7] the SO knowledge is stored in repository organized in function of the SO role, while in [8] relational databases record SO-related information. In the ACOSO-based SO model the information is spread between the KBManagementSystem and the agent behavior.

III. Simulation-based Analysis

As highlighted in the previous Section, the ACOSO-based SO model, leveraging on both the agent and MAS concepts, enables the effective modeling of SO-based IoT networks at different development phases. In this Section, the simulation of IoT networks allowing the validation of models, protocols and algorithms before the actual deployment of the network infrastructure, is addressed. Indeed, IoT networks simulation is an important but complex task because, depending of different scales, the number of the SOs may vary from dozens (e.g. home automation or body sensor networks) to thousands (e.g. Smart City scenario), with a different degree of density and different communication paradigms depending on the specific service. In addition, factors unrelated to the applications but specifically associated to the networking (e.g. traffic congestion, wireless signal attenuation and coverage, etc.) influence the SOs interactions and the service provision/fruition. Taking into account these issues and particularly focusing on communication among SOs, the IoT networks previously described through the agent-oriented approach are simulated by means of Omnet++ [5]. The modeling of an agent through an Omnet++ network node is straightforward. In fact, each network node/SO can be considered as an autonomous agent whose behaviors and tasks, which realize SO decision making and service provisioning (see Table I), are implemented at the application layer. All the other tasks related to transport-network-link protocol implementations, wireless connectivity issues, physical environment modeling are carried out by Omnet++.

In particular, Omnet++ makes it possible to design nodes with different communication boards or interface ports (to simulate the connection with external physical devices); in the following simulations, due to the intrinsic wireless nature of the IoT networks, nodes with 802.11 wireless boards have been considered.

Simulations have inspected IoT networks in the Information Exchange phase (IE) by exploiting TCP-based reliable and UDP-based unreliable transport protocols. The round trip time (RTT) and the packet delivery ratio (PDR) have been hence measured considering nodes that exchange empty messages and exploiting either a Client/Server (C/S) or a Peer-to-Peer (P2P) paradigm. Deterministic (1 pk/s) and stochastic Normal (with 0.5 mean and 0.2 variance) data generation models (DGM) have been used. In Table II the communication settings exploited in the simulations for the RTT/PDR measuring are shown.

Table II
Information Exchange (IE) Settings

Parameters	DGM (Data Generation Model)
Patterns	P2P (Peer-to-Peer), C/S (Client/Server)
Protocols	R (Reliable), U (Unreliable)

Both RTT and PDR have been evaluated in the context of small-, medium-, large-scale IoT networks with different SOs density. In particular, simulations took into account:

- The number of involved SOs (#SOs), since network congestion may increase depending of the SOs population. In the following SOs population is considered limited to 100 nodes for small-scale networks, 500 nodes for medium-scale networks and 1000 nodes for large-scale networks;
- SO distribution in a different number of subnetworks (#subnetworks). In the following it is assumed that small-scale networks are constituted by a single network, medium-scale networks comprise two or more subnetworks deployed in the same area so that their coverages overlap, and finally large-scale networks comprise two or more subnetworks deployed in not adjacent area;
- The deployment area in which SOs are located (supposing that they have no mobility patterns) since the proximity of multiple SOs may cause signal interferences in the wireless communications. For sake of simplicity, square grid areas with different side dimensions have been considered.

Table III summarizes the scenarios tested during the simulation phase.

Table III
Simulation Scenarios

	#SOs	# subnetworks	Grid side [m]
Small Scale (S)	5..100	1	5..100
Medium Scale (M)	20..500	1..10	From 100
Large Scale (L)	20..1000	1..20	From 100

In the following, for the sake of space, only some results of the simulations are shown, and in particular the PDR in the case of small-scale networks with #SOs increasing (see Fig. 1) and the RTT in the other two scales with #subnetworks increasing (see Fig. 2 and 3).

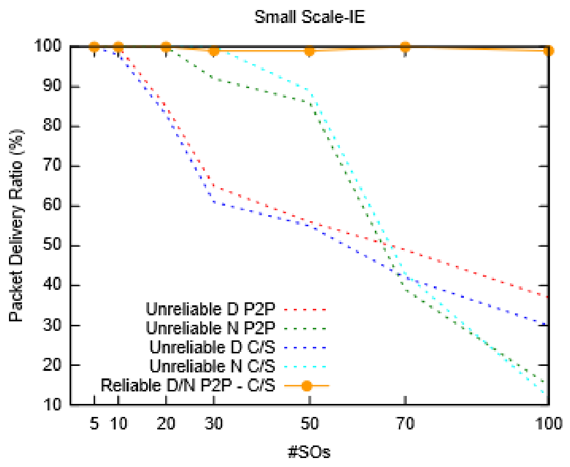

Figure 1: PDR when the #SOs increases in a square grid of side 100 m

Fig. 1, in the case of unreliable protocol shows that (i) with increasing of SOs, PDR decrease due to communication fails caused by interferences; (ii) with the same DGM, C/S and P2P patterns are equivalent; (iii) with a small number of SOs (less than 70) non-deterministic data generation models outperform the deterministic ones, while with the increase of SOs the trend inverts. In the case of reliable protocol, as might be expected, the PDR keeps the maximum value regardless of the DGM, patterns, and #SOs.

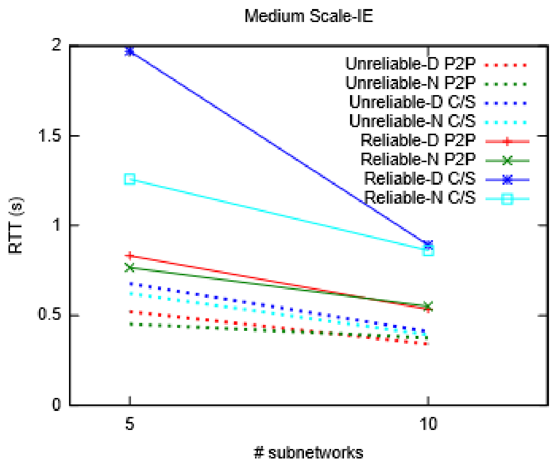

Fig. 2: RTT of 500 SOs equally distributed into 5 and 10 subnetworks

Fig. 2 shows that the RTT decreases if the same number of SOs (500) is deployed on the same area but distributed on more subnetworks (5 and 10 square grids of side 100 m), since the traffic is well-balanced. As might be expected, the unreliable protocols outperform the reliable ones.
Fig. 3 shows that in the large-scale scenario (5, 10 and 20 subnetworks deployed in a squared area of side 100m with 50 SOs each) the absence of interferences among the subnetworks generates RTT values quite stable and lower than the correspondent ones in the medium-scale scenario.

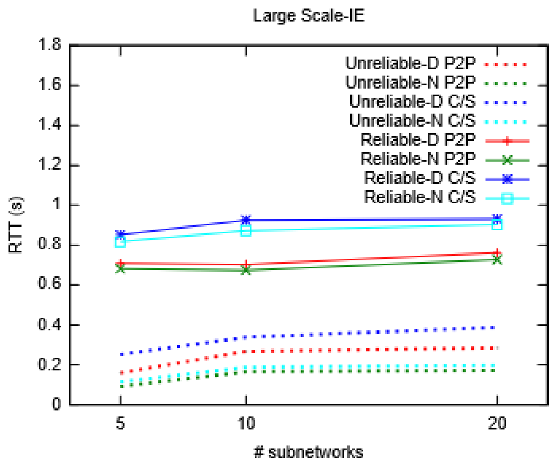

Fig. 3: RTT of 5,10 and 20 subnetworks of 50 SOs each

IV. Conclusion

In this paper the agent-oriented modeling of SO systems through the ACOSO-based SO model has been presented, by considering that (i) SOs and software agents share multiple features, and (ii) agent-based modeling flexibly assists the conceptualizing of dynamic and autonomous distributed systems in different contexts. Beside the agent-oriented SO modeling, IoT networks of different scales have been simulated through the Omnet++ simulator, with a particular attention to the SO communication in order to evaluate performances (focusing on bottlenecks, issues and networks dynamics) and validate network design choices. Simulation results highlight that multiple factors influence the network performance (as the SO number, their more or less dense distribution and the selected communication settings) and thus their combination should be made by following an application-driven approach.

Further research efforts will be devoted to (i) provide a full mapping between ACOSO-based SO model and Omnet++ simulation, in order to translate on such simulator the behavioral/application side of the SO systems, including also the simulation of the device layer; (ii) develop a full-fledged methodology for the analysis, design, simulation, implementation and validation of SO-based IoT systems [14, 15].

References

[1] G. Kortuem. et al. “Smart objects as building blocks for the internet of things,” *Internet Computing*, IEEE, vol. 14, no. 1, 2010, pp. 44-51.

[2] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” *Computer networks,* vol. 54, no. 15, 2010, pp.2787-2805.

[3] W. Van der Hoek, and M. Wooldridge, “Multi-agent systems,” *Handbook of Knowledge Representation,* 2008, pp. 887-928.

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

[5] A. Varga, *OMNeT++. Modeling and Tools for Network Simulation*, Springer Berlin Heidelberg, 2010, pp. 35-59.

[6] C. Macal, and M. J. North, “Tutorial on agent-based modeling and simulation,” in *Proceedings of the 37th conference on Winter simulation*, 2005, pp. 2-15.

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

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

[9] 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.

[10] G. Fortino, A. Guerrieri, G. M. P. O'Hare, and A. G. 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. 1934-1952.

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

[12] F. Bellifemine, G. Fortino, A. Guerrieri, and R. Giannantonio, “Platform-independent development of collaborative Wireless Body Sensor Network applications: SPINE2,” *Systems, Man and Cybernetics (SMC 2009), IEEE International Conference on*, 2009, pp. 3144-3150.

[13] 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.

[14] G. Fortino, A. Garro, and W. Russo, “Achieving Mobile Agent Systems interoperability through software layering,” *Information & Software Technology*, vol. 50, no. 4, 2008, pp. 322-341.

[15] G. Fortino, A. Garro, and W. Russo, “An integrated approach for the development and validation of multi-agent systems,” *International Journal of Computer Systems Science & Engineering*, vol. 20, no. 4, 2005, pp. 259-271.