Objects that Agree on Task Frequency in the IoT: a Lifetime-Oriented Consensus Based Approach

Giuseppe Colistra, Virginia Pilloni, Luigi Atzori
DIEE, University of Cagliari, Italy
{*giuseppe.colistra,virginia.pilloni,l.atzori*}*@diee.unica.it*

***Abstract*—Some key features of the end-systems impact on the way communications happen within the IoT: available objects' resources are limited, different objects may provide the same information (e.g. sense the same physical measure), the number of nodes in the IoT is quickly overcoming the number of Internet hosts with greater reliability issues. This entails for a new paradigm of communication with respect to those characterizing the traditional Internet. Before providing the required information about the physical world, objects coordinate with the other objects in *groups* and provide a unified service to the external world (the application that requires the service), with the intent to distribute the load of the requested services according to specific community defined rules, which could be: lifetime extension, QoS (Quality of Service) maximization, reward maximization, or others. In this paper other than describing the characteristics of this new communication paradigm and challenges it is called to address, we also propose a first solution for its implementation that relies on a distributed optimization protocol based on the consensus algorithm. Results of simulations and real experiments are also presented that show the viability in implementing the new communication model in a distributed way.**

Keywords*—*Consensus, resources allocation, Internet of Things.

I. Introduction

The last few years have been characterized by the technological revolution of the Internet of Things (IoT) [1], which represents a vision where devices with different capabilities (e.g., sensors, actuators, Radio Frequency Identification tags, smartphones, embedded devices) are reachable through the Internet in order to augment the current digital world with physical world information. As we know since its invention, the Internet interconnects nodes with dissimilar characteristics without central authorities by introducing some simple yet effective protocols that allow for nodes' interoperability so that information is successfully exchanged and services are provided by servers to clients and among peers. Fortunately, the same happens among objects in the IoT so that interoperability is assured and the data sensed by objects distributed and connected to the physical world is now available for the benefit of the human users.

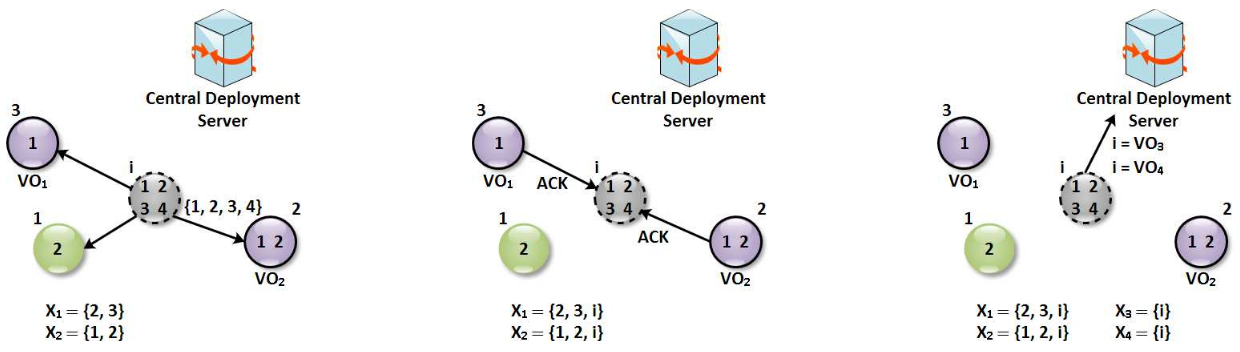
Some key features however characterize many IoT objects: i) available nodes' resources (electrical energy, memory, processing, node capability to perform a given task) are often limited. This is the case, for example, of battery powered nodes, which have limited energy amounts. ii) sensors may provide information that is not unique but can be generated by set of different objects which for example are capable to sense the same physical measure of the same geographical. iii) the number of nodes in the IoT is quickly overcoming the number of hosts in the ́traditional' Internet and most of these have a low reliability due mostly to the mobility and energy. This entails for a new paradigm of communication according to which objects coordinate with the other objects in *groups* and provide a unified service to the external world (the application that requires the service), with the intent to distribute the load of the requested services according to specific community defined rules, which could be: lifetime extension, QoS (Quality of Service) maximization, reward maximization, or others. It is evident that an appropriate coordination of objects' resources utilization would consistently improve their performance. Given the size of a distributed heterogeneous system such as the IoT network, the optimal creation of communities and the resource allocation within are not trivial issues. Furthermore, typical IoT networks are characterised by the dynamic behaviour of their nodes. In fact, emerging applications in smart environments such as smart cities and smart homes, where IoT is preponderant, are often based on opportunistic networks. In opportunistic networks, connections among nodes are created dynamically in an infrastructure-less way: when forwarding a message, next hops are chosen opportunistically, on the basis of their likelihood to get the message closer to its destination [2]. In such a dynamic context, with frequent and quick changes of scenario, it is not reasonable the community management to be centralized but a distributed approach has to be followed.

The major contribution of this paper is twofold: we first describe the characteristics and challenges that the proposed new communication paradigm is called to address (in Section III); we then propose a first solution for its implementation which relies on a distributed optimization protocol based on the consensus algorithm (described in Section IV), which solves the problem of resource allocation and management preserving the required QoS. The paper is completed with Section II that analyses some past works, Section V that provides a performance analysis, and Section VI that draws final conclusions.

II. Past Works

Resource allocation has been extensively studied in the Wireless Sensor Network (WSN) field [3]. A big effort has been put into resource allocation to extend WSN lifetime. In

This work has been partially funded by the project Artemis JU Demanes, Design, Monitoring and Operation of Adaptive Networked Embedded Systems, grant agreement no. 295372.



(a) Node i joins the network

(b) VO_1 and VO_2 send ACK to node i

(c) Node i becomes VO_3 and VO_4

Fig. 1. Sequence of steps when a new node i joins the network

[4] the authors propose a framework to estimate network performance, which doesn't use node resources and does not introduce additional energy consumption that could compromise the network lifetime. In [5] a centralized task allocation algorithm to improve network lifetime is proposed. This study focuses on the reduction of the overall energy consumption into a heterogeneous WSN, with attention to nodes' residual energy. In [6] the same problem is analysed taking into account also task execution time. The authors in [7] propose the DLMA, an overlaying framework that determines the distribution of tasks among the nodes in a WSN by means of a distributed optimization algorithm, based on a gossip communication scheme, aimed at maximizing network lifetime. A similar approach is studied in [8], where a distributed algorithm based on particle swarm optimization is proposed.

As far as IoT networks are concerned, resource allocation is an open issue. Network heterogeneity, which regards both node capabilities and characteristic parameters, makes the resource allocation a challenging task. Semantic descriptions are needed, so that a common middleware can be designed in order to ensure interoperability among different devices. Comprehensive ontologies that provide a semantic model for IoT are defined in [9] and [10]. Most of the existing studies on resource allocation for IoT are focused on IoT service provisioning, such as in [11] and [12]. In these studies, the aim is to find and allocate the resources that enable service execution. However, they do not focus on finding the best configuration that corresponds to an optimal resource allocation. None of the works found in the literature tries to find the optimal resource allocation associated to the lowest impact of the application assigned to the network. In this paper, a distributed optimization protocol, based on consensus algorithm [13], that solves the problem of resource allocation and management is described. Consensus algorithms consist in a collection of laws that regulates the interaction and the exchange of information between each node and its neighbours (nodes that are only one hop far from each other). They have been used to a great extent in problems involving interconnection of dynamic systems, such as flocking theory [14], rendezvous in space [15], and distributed sensor fusion in sensor networks [16].

III. Architecture

In typical IoT scenarios it frequently happens that some nodes perform the same task, such as the measurement of the traffic in the same street, the detection of moving objects/persons in a given environment, or the processing of measured data. Accordingly, the IoT is made of groups of nodes, i.e. *task groups*, that perform similar and replaceable tasks. To understand the meaning of *task group*, suppose, for example, that the network performs a temperature sensing in an area called A: only those nodes that are equipped with a temperature sensor and that are deployed within area A are included in the *task group* related to this task.

The scenario proposed in this paper is that of an opportunistic IoT, where nodes continuously join and leave the network. In this scenario, the IoT is made of *virtual objects* (VOs) [17] that are activated by the Central Deployment Server, which is in charge of identifying the objects tasks that have to be activated to deploy the target application. The VO role may be implemented by a node in the *task group* and is in charge of processing the requests generated by the central server and forwarding configuration messages to the other physical nodes (note that the VO may coincide with the only single physical node that is capable of implementing the required task). Then, the nodes in the *task group* autonomously decide how to properly allocate resources to the required task by distributing the burden among them. The aim of the algorithm explained in the following is to dynamically change tasks' assignment to share the effort required to perform the considered task, in terms of necessary network resources. More precisely, nodes involved in the same task, i.e. belonging to the same *task group*, converge to the same lifetime by adjusting the frequency at which they perform the task. In this way, the resources required to complete the task are equally distributed among nodes, and the network lifetime is, therefore, improved.

In order for nodes to start a negotiation, they need to have already joined the related *task group*. This join procedure will now be explained in detail. As soon as a node i joins the network, it broadcasts to its one-hop neighbours the information related to the tasks that it is able to perform. Accordingly, each VO related to task k, namely VO_k, adds node i to the list of nodes X_k that belong to its *task group*, and replies with an acknowledgment. If no VO is associated to one or more tasks yet, node i is designated as VO for those tasks, and it sends this information to the Central Deployment Server. As shown in Fig. 1, suppose, for example, that node i is able to perform 4 tasks. Thanks to the middleware running on top of the network, the description of those tasks is converted to 4 different ID numbers, corresponding to the IDs of the equivalent *task groups*. Suppose that, in this example, node i is associated to *task groups* $\{1, 2, 3, 4\}$, the VOs VO$_1$ and VO$_2$ are associated to *task groups* 1 and 2, and that no VO has been associated to *task groups* 3 and 4, yet. Thus, VO$_1$ and VO$_2$ will add node i to their list of nodes X_1 and X_2. Obviously, node i will not receive acknowledgements for *task groups* 3 and 4. Hence, node i will assume the role of both VO$_3$ and VO$_4$, and it will inform the Central Deployment Server.

When an application requires the execution of a given task, the Central Deployment Server sends an activation signal to the appropriate VOs, which forward it to their list of nodes. Then, the negotiation algorithm is started. The nodes of the involved *task group* start to adjust their frequency of task execution, in order to distribute the task load among themselves. The algorithm ensures that the required QoS is always preserved. In fact, the total number of samples per second collected by the *task group* is constrained to be constant. In the following, this negotiation algorithm will be described in detail.

IV. Consensus among task group nodes

A. Agreement on task frequency among nodes

Each node i that performs task k collects data with frequency $f_{i,k}(t)$. The power consumed by node i to perform task k is expressed by:

$$P_{i,k}(t) = E_{i,k} f_{i,k}(t) \tag{1}$$

where $E_{i,k}$ is the energy spent by node i for task k. Assuming to keep the total number of samples per second generated by all the nodes in the *task group* constant and equal to the application requirement F_k, $f_{i,k}(t)$ can be changed by nodes within the same *task group* to find the optimal combination that distributes the task burden among them. Energy $E_{i,k}$ can be split in different contributions such as sensing energy $E_{i,k}^{sens}$, transmission energy $E_{i,k}^{tx}$ and computing energy $E_{i,k}^{comp}$:

$$E_{i,k} = E_{i,k}^{sens} + E_{i,k}^{tx} + E_{i,k}^{comp} \tag{2}$$

Each node i is also associated to a residual energy $E_i^{res}(t)$ that depends on its residual battery charge – and thus decreases with time –, and on its lifetime $\tau_i(t)$, which is the time before running out of battery. Node i's lifetime is expressed as:

$$\tau_i(t) = \frac{E_i^{res}(t)}{\sum_k P_{i,k}(t)} \tag{3}$$

In the proposed solution we aim at maximising the network lifetime, which is the time before the first node fails. This objective is equivalent to the target of having uniform objects' lifetimes in the community. Considering only task k, we define the contribution in lifetime of node i to task k as:

$$\tau_{i,k}(t) = \frac{E_i^{res}(t)}{P_{i,k}(t)} = \frac{E_i^{res}(t)}{E_{i,k} f_{i,k}(t)} = \frac{1}{c_{i,k}(t) f_{i,k}(t)} \tag{4}$$

In order for nodes to tend to a uniform lifetime, this contribution should tend to an amount τ_k that is equal for all the nodes involved in task k's execution, i.e.

$$\lim_{t\to\infty} \tau_{i,k}(t) = \tau_k \qquad \forall i \in X_k \tag{5}$$

To lighten the notation, in Eq. 4 we add $c_{i,k}(t)$ as a cost function that reflects the extent to which the node can be used, according to the ratio between its energy consumption for task k and its residual energy. At the same time t if $c_{i,k}(t) < c_{j,k}(t)$, i can be used more extensively than j.

For a graphical representation of the problem at a time t, we can draw the residual energy $E_i^{res}(t)$ and the consumed power $P_{i,k}(t)$ on the x and y axes, respectively. As shown in Fig. 2, mapping all combinations for all the nodes, we obtain a constellation of points where each node has a different lifetime $\tau_{i,k}(t)$, which depends on the initial node battery status (yellow points in Fig. 2). By applying Eq. 5 we force these points to move into a a straight line with a slope τ_k by changing each frequency $f_{i,k}(t)$, so that each node will have the same lifetime (white points in Fig. 2). The required changes are driven by the community QoS target expressed in terms of the total number of samples per second provided by the community F_k, so that the following constraint is introduced:

$$\sum_{i=1}^{N_k} f_{i,k}(t) = F_k \tag{6}$$

When the appropriate VO forwards an activation signal to the list of interested nodes, the message conveys F_k. We can apply Eq. 4 into Eq. 6 as follows:

$$\tau_k(t) = \frac{1}{F_k \sum_{i=1}^{N_k} c_{i,k}(t)} = \frac{1}{F_k N_k \overline{C_k}(t)} \tag{7}$$

To reach the goal each node in the task group needs to know $\overline{C_k}(t)$, i.e. the mean value of all the $c_{i,k}(t)$ values for task k. After a node knows this value, it can evaluate the frequency that corresponds to its optimal lifetime:

$$f_{i,k}(t) = \frac{1}{c_{i,k}(t)\tau_k(t)} = \frac{F_k N_k \overline{C_k}(t)}{c_{i,k}(t)} \tag{8}$$

To compute the value of $\overline{C_k}(t)$ the numerosity of the *task group* is needed. This is possible because the VO has the list of interested nodes, so N_k can be forwarded at the beginning of the process by means of the activation signal. If the conditions of the network change during the task execution (e.g., a new node enter the community or a node fault happens), the nodes detecting the change flood the message so that each node is reached. A centralized solution is computationally very simple, but with respect to a decentralized solution, it requires higher transmission costs due to a lot of control messages and the system has a slower reactivity due to topology or node status (i.e. residual energy) changes.

B. The distributed solution based on the consensus protocol

To reach the goal, in a totally distributed way, an average consensus protocol is implemented by the nodes within the *task group* to evaluate $\overline{C_k}(t)$. We focus on a particular class of iterative algorithms for average consensus. To estimate the average value across the network, we use a consensus protocol propagation that is totally asynchronous and distributed. When the procedure starts, each node allocates a variable $\overline{c_{i,k}}(t)$ to iteratively estimate the value of $\overline{C_k}(t)$. At an initial time $t = 0$:

$$\overline{c_{i,k}}(0) = c_{i,k}(0) \qquad \forall i = \{1, \cdots, N_k\} \tag{9}$$

In order for the estimated $\overline{c_{i,k}}(t)$ to converge towards the correct average $\overline{C_k}(t)$ (computed on all nodes in the *task group*), each node follows the rule of the consensus protocol and updates the local estimation by adding a weighted sum of the local discrepancies, i.e., the differences between neighbouring node estimated values and its own. At each time step $t+1 > t$, the update rule of the consensus protocol is the following:

$$\overline{c_{i,k}}(t+1) = \overline{c_{i,k}}(t) + \sum_{i=1}^{O_j} W_{i,j}(\overline{c_{j,k}}(t) - \overline{c_{i,k}}(t)) \tag{10}$$

where O_i defines the number of node j neighbours.

$W_{i,j}$ is a weight associated with the communication between i and j. If the weights are associated with undirected edges, we have $W_{i,j} = W_{j,i}$. It's also possible to consider asymmetric weights, associated with ordered pairs of nodes, so the previous equality is not true. Weights have to satisfy some basic constraints, as well as the convergence condition, such as defined by Xiao et al. in [18]. So we decide to take very simple set of weights that define a one hop communication (the node i communicates only with the neighbours):

$$W_{i,j} = \begin{cases} 1 & if \quad j \in O_i \\ 0 & if \quad j \notin O_i \end{cases} \tag{11}$$

From the described protocol follows that the $i-th$ node transmits only the value of $\overline{c_{i,k}}(t)$ and subscribes this value each iteration with the neighbour. So the protocol does not rely on extensive transmissions and the related amount of data exchanged is very small. From the point of view of the node buffer occupancy, each node stores only one variable and it subscribes this at each iteration with the neighbours, so the memory required by the protocol is very limited.

V. Performance Analysis

The performance analysis focuses on two case studies: the first one is a simulated scenario; the second one is a real scenario. To better understand the problem and to simplify the analysis we consider a scenario where the residual energy decreases very slowly than the convergence of protocol, so we can consider all terms as time independent.

A. Simulation Scenario

In this case study we used the Matlab software to implement a framework to simulate the protocol using broadcast communications among the nodes. The network topology has been created following a random geometric distribution and transmissions on the network are asynchronous. The broadcast communication entails that if i sends a packet, this is received by all neighbours, which update their values. The simulation was run on 20 nodes (i.e. $N_k = 20$) in a random topology. We initialized nodes values: $E_i^{res}(t)$, $E_{i,k}$, $f_{i,k}(t)$ with random values, the QoS request F_k has a random value, too. We assumed that each node transmit 500 packets. By the simulation we intended to study the performance of the protocol in terms of convergence speed and error, considering a broadcast communication among nodes. Fig. 2 shows the protocol lifetime convergence. Yellow points show the initial condition of network nodes. After 20 packets transmitted on the network, each node has corrected its task frequency and tends to the convergence (red points). Green points show the state of the nodes after 80 iterations. In the end, white points show the final state of the node, after 500 iterations. At the end of the simulation the protocol can be considered converged. In fact, in Fig. 2 white points are very near the ideal position that is marked by the blue line. So by Fig. 2 is clear that as the number of exchanged packets increases, nodes reach a better consensus and, in this case, the same lifetime.

From the QoS point of view, we analyse the percentage error with respect to the QoS constraint F_k. Fig. 3 shows that the initial error is -25%, but before 50 packets are transmitted this value decreases by 10%, and after 50 iterations we obtain a very low error value of about 5%. So it's possible, after a first transition time, to consider also the QoS constraint satisfied.

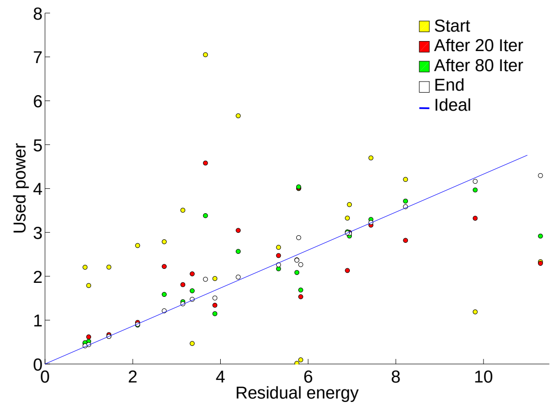


Fig. 2. Lifetime consensus

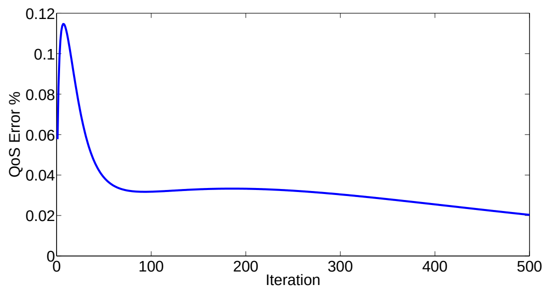


Fig. 3. QoS error

B. Real Scenario

The last experiment consists in the study of the protocol performance in a real scenario. In this section, we firstly illustrate the tools that we used. Then, we analyse the results to validate the performance of the proposed consensus application. The tools used for the experiments are the following:

- Development kit case provided by Telit Wireless Solutions. This kit is made of ZigBee radio boards that are based on the CC2530 Chip with the Embedded Telit Z-One ZigBee-PRO Stack. The antennas are external dipoles characterized by an omnidirectional pattern.
- The software Wireshark is used to inspect the packet content. To analyse the performance of the network from

the Wireshark output and to conduct network discovery and commissioning, a specific tool named SRManager Tool has been developed by Telit Wireless Solution in collaboration with our lab. In this experiment, this tool has been used to set up the consensus protocol experiments.

During the experiments we used three devices that communicated using the ZigBee standard on channel number 14 in the 2.4 GHz ISM frequency band and we used one device in sniffer mode to capture the packets on the network. The type of communication is the gossip. This modality entails that two nodes communicate to update their values, instead the broadcast when each node communicate with all neighbours. To reduce communication overhead, we inserted the necessary information inside the overhead of the packets.

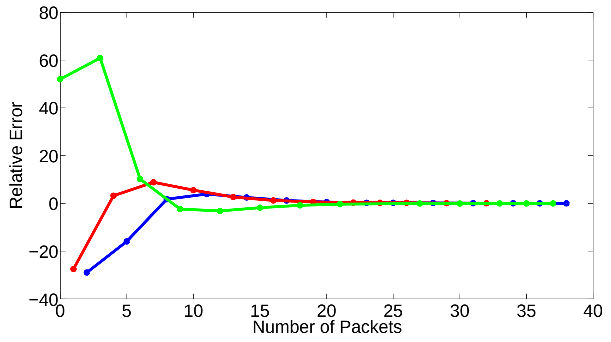


Fig. 4. Convergence error in real scenario

Fig. 4 shows the algorithm convergence in real scenario. As in the simulation discussed in the previous subsection, nodes converge at consensus. A good consensus has been reached after about 15 packets exchanged, corresponding to a mean of 5 update for node. From the error point of view the initially error reaches peaks of 60% and −30%. Nevertheless, after 15 packets are transmitted, this value decreases by ±20%, and eventually we obtain a very low error value of about ±3%.

VI. Conclusions

In this work, we presented a new consensus protocol for a distributed decision on resource allocation in a IoT scenario with the aim to improve the network lifetime and management preserving the required Quality of Service (QoS). We simulated a IoT scenario with many heterogeneous nodes involved in the same task. The simulated case convergence has shown to be very fast. The proposed protocol allowed for improving the lifetime network because each node's lifetime tend to an amount s that is equal for all the nodes involved in the same task. In terms of preserving the required QoS the proposed protocol allowed for achieving an error lower than 5% than the QoS constraint. Respect to a centralized solution, the proposed protocol is totally decentralized and asynchronous, so it brings benefits such as reduction of control message transmissions and increase in system reactivity. Real experiments validated the simulation results. Node lifetime proved to converge and the percentage error was lower than ±20% after an average of 5 packets transmitted by each node, and lower than ±3% after 30 packets.

Future works will be focused on the extension of the protocol to the multitasking optimization. Furthermore, the proposed resource allocation protocol can be easily extended in a scenario with a network that changes topology quickly.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787 – 2805, 2010.

[2] C.-M. Huang, K.-c. Lan, and C.-Z. Tsai, "A survey of opportunistic networks," in *Advanced Information Networking and Applications-Workshops, 2008. AINAW 2008. 22nd International Conference on*. IEEE, 2008, pp. 1672–1677.

[3] K. L. Mills, "A brief survey of self-organization in wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 7, no. 7, pp. 823–834, 2007.

[4] G. Colistra and L. Atzori, "Estimation of physical layer performance in wsns exploiting the method of indirect observations," *Journal of Sensor and Actuator Networks*, vol. 1, no. 3, pp. 272–298, 2012.

[5] V. Pilloni and L. Atzori, "Deployment of distributed applications in wireless sensor networks," *Sensors*, vol. 11, no. 8, pp. 7395–7419, 2011.

[6] J. Zhu, J. Li, and H. Gao, "Tasks allocation for real-time applications in heterogeneous sensor networks for energy minimization," in *Proceedings of the Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, vol. 2, 2007, pp. 20–25.

[7] V. Pilloni, M. Franceschelli, L. Atzori, and A. Giua, "A decentralized lifetime maximization algorithm for distributed applications in wireless sensor networks," in *Communications (ICC), 2012 IEEE International Conference on*, 2012, pp. 1392–1397.

[8] Y. Shen and H. Ju, "Energy-efficient task assignment based on entropy theory and particle swarm optimization algorithm for wireless sensor networks," in *Green Computing and Communications (GreenCom), 2011 IEEE/ACM International Conference on*, 2011, pp. 120 –123.

[9] W. Wang, S. De, R. Toenjes, E. Reetz, and K. Moessner, "A comprehensive ontology for knowledge representation in the internet of things," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, 2012, pp. 1793–1798.

[10] S. De, T. Elsaleh, P. Barnaghi, and S. Meissner, "An internet of things platform for real-world and digital objects," *Scalable Computing: Practice and Experience*, vol. 13, no. 1, 2012.

[11] D. Guinard, V. Trifa, F. Mattern, and E. Wilde, "From the internet of things to the web of things: Resource-oriented architecture and best practices," in *Architecting the Internet of Things*. Springer, 2011, pp. 97–129.

[12] B. Silverajan and J. Harju, "Developing network software and communications protocols towards the internet of things," in *Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE*. ACM, 2009, p. 9.

[13] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[14] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *Automatic Control, IEEE Transactions on*, vol. 51, no. 3, pp. 401–420, 2006.

[15] V. Terziyan, O. Kaykova, and D. Zhovtobryukh, "Ubiroad: Semantic middleware for context-aware smart road environments," in *Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on*. IEEE, 2010, pp. 295–302.

[16] M. Kuorilehto, M. Hannikainen, and T. Hamalainen, "A middleware for task allocation in wireless sensor networks," in *Personal, Indoor and Mobile Radio Communications, 2005. PIMRC 2005. IEEE 16th International Symposium on*, vol. 2, 2005, pp. 821–826 Vol. 2.

[17] J. Pascual Espada, Ó. Sanjuán Martínez, G. Pelayo, C. Bustelo, and J. M. Cueva Lovelle, "Virtual objects on the internet of things," *IJIMAI*, vol. 1, no. 4, pp. 23–29, 2011.

[18] L. Xiao, S. Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33 – 46, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0743731506001808