Simulating BDI-based Wireless Sensor Networks

Alexis Morris, Paolo Giorgini, Sameh Abdel-Naby
Department of Information Engineering and Computer Science
University of Trento
Trento, Italy
{alexis.morris, paolo.giorgini, sameh}@disi.unitn.it

***Abstract*—"Autonomic systems" merge advancements in the field of multi-agent software design, dynamic analysis, and decentralized control in order to assist designers in constructing complex distributed systems. Wireless Sensor Networks (WSN) represent such systems, and may benefit from autonomic system designs that target distributed nodes in diverse and changing environments that interact over a wireless communication channel for decentralized problem solving. Multi-agent system techniques have been recently applied to WSN's; however, due to hardware limitations nodes (agents) are not fully deliberative (or strong) reasoning systems. Since hardware increases rapidly it is expected that such systems may eventually be viable. In this paper we provide a generic, extensible, and deliberative simulator for testing interactions in autonomous WSN's. The belief, desire, intention (BDI) agent model of Rao is used, as well as the Agentspeak language, and the Jason framework. Results from two simple WSN test scenarios show how (simulated) BDI agents might perform basic WSN functions.**

***Keywords*-Wireless Sensor Networks; Distributed Systems; Autonomous Systems; Belief Desire Intention; Simulation;**

I. Introduction

Wireless sensor networks (WSN's) are teams of small, low powered computing devices that measure physical factors of the environment, sharing information across nodes via wireless communication channels. As the hardware for the sensing units becomes smaller, and more power efficient, it is expected that WSN's will be embedded into many everyday objects in order to both gather data and perform pervasive computing tasks [21] as part of a future "internet of things". Given such a ubiquitous context, complexity of design, deployment, and runtime interaction becomes more problematic; especially since the dynamic nature of the many varied problem domains, and the limited local intelligence of sensor devices makes it difficult to deploy even a simple WSN. Testing of deployments is also problematic, as conventional methodologies are not able to compensate for the number of variables that the device may encounter. One solution to these problems is to use complicated, cleverly designed, distributed systems algorithms to make the system function more robustly. A more natural solution is to create an autonomous wireless sensor network that is able to react dynamically to changes when required, but in a decentralized manner.

Systems that use such an approach are flexible, adaptable, self organizing, and self-optimizing [12], allowing designers to solve distributed problems using interaction design policies rather than algorithms [25]. In the literature (see [24]), researchers are taking advantage of the approach in order to solve common distributed system problems such as routing, active sensing, information processing, and collaborative sensing strategies. The autonomous WSN as a platform is shown to achieve improved system longevity, [19], and coverage density [26], two of the most important WSN problem areas. The common feature of current agent-oriented approaches is that they employ non-deliberative agents; those that do not have a fully functioning reasoning system. These agents use a weaker notion of agency than the belief-desire-intention (BDI) of Rao, [17], that is commonly held to be the standard model of strong agency [10]. What this means is that such agents are not as reactive to the environment as they could be; leaving an open problem for finding out what gains may be obtained through deliberative agent-oriented WSN's.

Section 2 describes the autonomic computing aspects of a WSN, as a motivating factor. Section 3 discusses the literature regarding non-deliberative and deliberative simulations of WSN's. Section 4 presents a proposed simulator for BDI-driven WSN development. Section 5 describes experiments for basic WSN tasks using BDI agents. Section 6 provides an evaluation of the simulator. Section 7 concludes the paper.

II. The Autonomous Nature of WSN Systems

WSN's in theory are not difficult to represent using multiagent system designs. Like agents, they must interact with each other, through communication, or coordination in the presence of potentially limited resources. The WSN is inherently dynamic in terms of structure, deployment settings, and behavior effects. Nodes situated in a real world environment sample continuously and relay information to other nodes, according to directives, priority of the information, or other factors. Communication is broadcast based, typically in an inter-dependent, multihop routing fashion to save power, [11]. Additionally, WSN nodes do not have a large battery capacity, so sending, storing, and receiving messages are costly operations. WSN solutions must share resource costs, much like agent systems must distribute tasks. Nodes may also be heterogeneous in terms of the data they are collecting,their owners, objectives, and current position in the environment. This coincides with self-interested agents in a society with limited resources.

These factors make the WSN attractive for exploring MAS engineering, coordination, and negotiation protocols. Finally the WSN environment domains are very unpredictable, typically representing a complex world state where the number of critical choice points to perform effectively is potentially

very high (Chrisman, [6], considers this state a "robot's worst nightmare," a term that fits for WSN's as well). An agent that has combined internal states of model reactivity, and contingency anticipation are appropriate for such situations.

III. Related Work

The current literature shows a number of agent based wireless sensor network implementations ([9], [27], [19], [26], etc). This is growing, as the advantages of the autonomous WSN becomes more accepted.

This section details those approaches that simulate agents in WSN's, that embed an agent component and a hardware model into a common framework, including environment modeling. The first is the work of Kho and Jennings, [13], for adaptive sampling within the FloodNet project, as mentioned previously. Here the authors present a non-deliberative simulation tool known as DC-WSNS, in order to evaluate the algorithms. DC-WSNS provides modules for nodes, batteries, sensing capabilities, network stack, and an environment model to represent cloud coverage for solar panels. They do not, however, model the wireless communication channel, transmission details such as messaging, or propagation.

Another similar approach is that of Ruiz, et al, [20], using the MANNA project to model fire risk monitoring in a WSN. The work adopts the commonly used Network Simulator 2 (NS-2), [18], and its adaptor for Wireless Sensor Networking. NS-2 provides a simulation environment that maintains common MAC protocols, a transmission layer, network layer, a broadcast mechanism, and an application layer (for implementing MANNA policies). Energy consumption, and data delivery rates are the main metrics gained by the simulation. This also is a non-deliberative agent approach.

In the work of Rogers, et al, another simulation is presented for the domain of the GLACSWEB project [19]. The simulator details are not discussed, however the results show simulation of network formation topologies and displays the network as nodes are dying off, and network restructuring takes place, allowing for maintaining the network coverage size. This approach is policy driven, and uses non-deliberative agency.

In Ma, et al, [15], the authors briefly discuss exploring the use of BDI agents in WSN's, and show the result of a simple application for power management. In this work, as the agents battery consumption level increases, the agent adjusts the radio state, and lowers sampling rates as a simple approach to save energy. The simulation framework uses the deliberative AgentFactory Micro Edition (AFME), [7], combined with the common J-Sim WSN simulator, [23]. This work is very similar, but uses AgentFactory's belief-commitment-plans (BCP) approach, and J-Sim's more detailed hardware model.

Finally, in a recent work, Zboril, [27], presents mobile agents for the CrossBow platform [8], designing a BDI-like framework, and middleware, known as "MOTE". Using ALLL (Agent Low-Level Language) and a specification of the CrossBow hardware they present a simulator known as T-Mass. T-Mass is able to simulate agent behaviors for the platform, but does not focus on the environment model. The ALLL language is a combination of calculus approaches, but

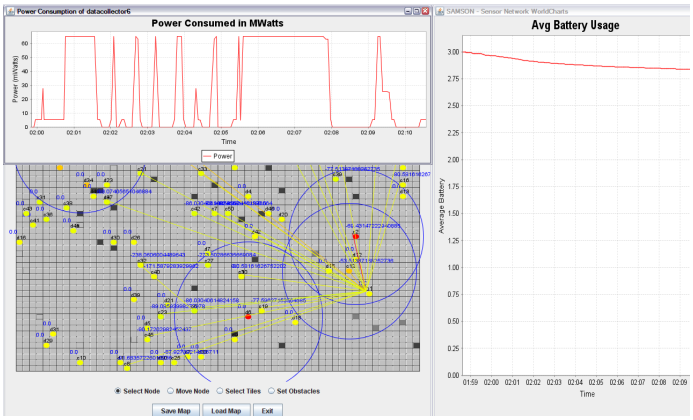

Fig. 1. The SAMSON WSN Simulator, [2], provides a Tileworld, sensors, dialogs, and power consumption charts.

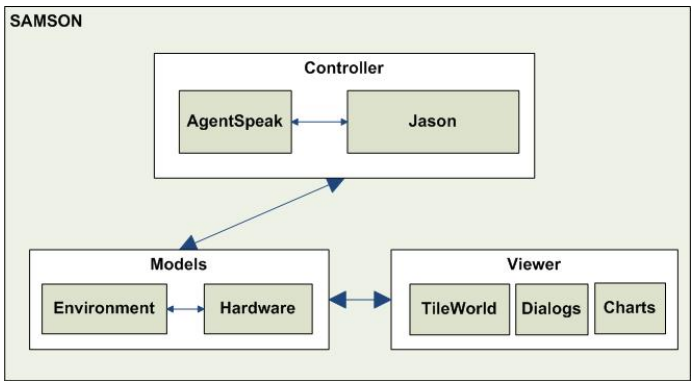

Fig. 2. The simulation architecture following the MVC pattern.

differs from Agentspeak's modal logic procedural reasoning system. Further the tool focuses on mobile agents that can share nodes on a virtual architecture.

IV. A Deliberative WSN Simulation Environment

In addition to a strong notion of agency, a complete solution for autonomous wireless sensor network simulation would require models of the environment, and hardware that are as accurate as possible, [14]. It would also be flexible enough for many WSN scenarios. Such a solution would consist of "model components", namely for power consumption, radio propagation, network management, packet management, radio channel model, and environment modeling. Additional utility would also be added through the possibility of adding an interpreter based on the TinyOS platform, or SystemC, or Java, allowing for easy translation of agent code into runnable WSN code. In this work, *a partial solution is presented as a first step towards BDI agency in a WSN*, based on the Agentspeak language, [16].

The proposed simulator, SAMSON, [2], is designed with the following in mind; it should have an Agent-based (BDI) controller and agent definition language; support for heterogeneous agents with different objectives and goals; scalability for testing variable sized agent networks; one to one mappings of agents to nodes; extensible model of hardware components, power, and radio; model of the environment (should include

node locations, and obstacles of various thickness); Java-based model for object-oriented design benefits; ease of programming agent logic; clean user interface, allowing for moving nodes and obstacles, and changing hardware parameters easily; visualizing charts of various properties (especially power consumption). Figure 2 shows the architecture of the proposed system design. The entire tool is constructed with the Jason/Agentspeak framework, [5], as the primary backbone. By using Agentspeak as the language, the tool is able to present heterogeneity and generic behaviors that are interpreted in a reactive fashion by the Jason interpreter. Each Jason agent has a corresponding node in the model, and access to that node's "hardware" through system percepts and actuator functions (see chapter 5 of [5] for more on how the Jason architecture communicates with custom models). The model of hardware and power specifications have been based on industry standard TMote Sky sensors, [22], [4].

In the model percepts are updated continually as part of the control loop of the interpreter, but also when a message is received by the Network component, or a sensor has retrieved data. This information is then acted upon by the agent, which is able to perform several preset, but easily customizable, actions. The main actions that are made available to agents are internal actions in Jason (see Table I). In particular, nodes may send messages, sense the environment, set hardware power and state, and conduct special actions for saving, and custom computation messages. Together, they allow an agent to control a model of a sensor, and designers of multi-agent systems to compose interaction protocols for different cases.

Action	Comment
sendTX(Msg)	Sends a message across the radio.
sense(SensorId)	Polls the sensor based on id.
setTXPower(Pwr)	Sets radio transmission power
setRadioState(State)	Sets radio status (8states)
setMCUState(State)	Sets micro-controller (3states)
saveData(Data)	Saves data to file
computePartner(Node, RCost, Pwr)	Special action for routing
setPartner(Node)	Special action for routing
resetPartner(Node, Pwr)	Special action for routing

TABLE I
ACTIONS AVAILABLE TO SAMSON AGENTS.

V. EXPERIMENTS

In order to test the simulator design, functionality, and usefulness, two main cases are used. The first presents a basic design for flooding in WSN's. Flooding is a classical data dissemination algorithm for wireless networks that is simple, and is useful for checking power computations and validating basic functionality, [3]. The second testcase is an industry study of a current WSN implementation by ArsLogica, [1], an Italian research and development company affiliated with the University of Trento. The company is largely interested in the simulator and its usefulness in early validation. Several demos of the simulation solution have been successfully conducted with the company during the course of the work.

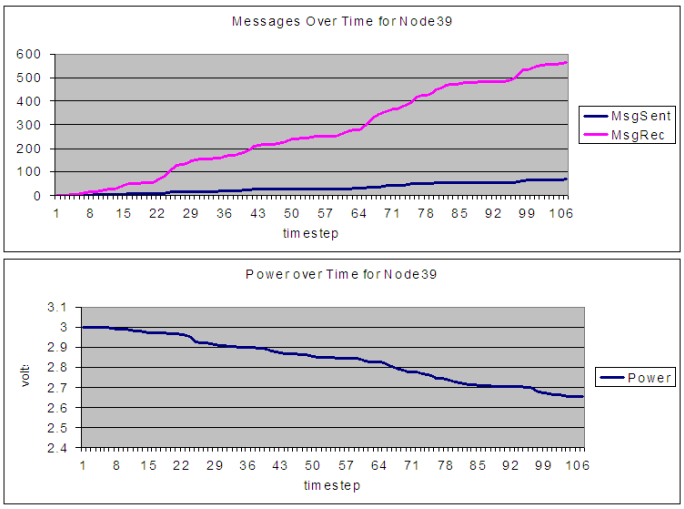

Fig. 3. Results gained from running a simple "Flood-hop routing" testcase show increased message receipts (and corresponding power losses) of a randomly chosen node in the network. This is typical flooding behavior. Flooding involves broadcasting any messages received to all neighbouring nodes; this approach is very wasteful in propagating duplicate messages.

The server management scenario represents a different kind of test, in that the focus is on reflecting the behavior of the real ArsLogica server implementation in terms of assessing what real nodes do, and how the agents correspond. Unfortunately, this case has no corresponding data from actual implementations, and thus only behavioral assessment is available. It is expected that the agents in this scenario will function as follows. Data collector nodes should perform low power listening, and thereby save power overall, while still forwarding data. Clusterheads should forward data to the base station via multihop routing, and reform relay partners after a request from the base station. This will result in higher amounts of message passing for clusterheads. The basestation only receives data and initiates net formation. Note that being able to reproduce this basic behavior will not show that agents improve the scenario, but instead that strong agency can be used for wireless sensor network interactions.

VI. EVALUATION

For the results shown the simulator performs partially well, providing a good sense of the flooding example, although it does not provide decent stopping conditions for the problem, hence messages tend to be continually sent. For the servers scenario, the simulation functions sufficiently to distinguish the behaviors of the three agent types; despite the discrepancy with the number of messages sent from data collectors not matching up with the number of received messages at the basestation. The multihop succeeded in transferring messages across the clusterheads to the basestation. In short, the agents managed to succeed in imitating some recognizable aspects of the system. As for the simulator itself, the results have shown that it is adaptable to different scenarios, with the five different agents discussed. Both test scenarios were able to reflect some of the real behavior, although the degree of realism is not directly quantified with actual data. The model for power consumption and message handling works as designed, but could

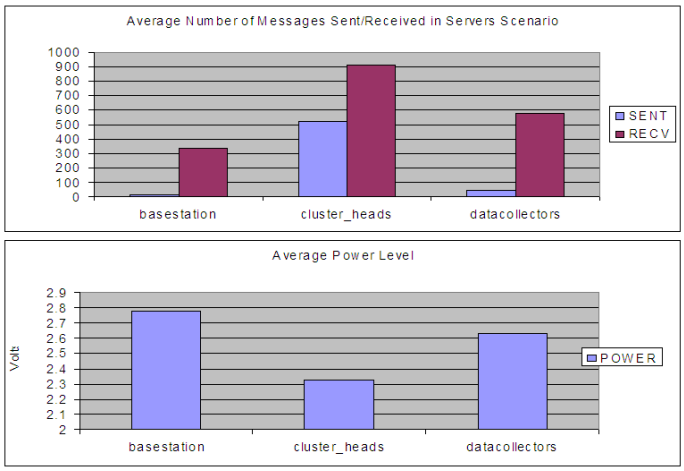

Fig. 4. Averages of messages sent and received in the Server scenario, and average power voltage remaining. Clusterheads sent the highest number of messages, and lost the most power, due to the multihop forwarding costs, combined with the costs of network reformation handling. The data collectors, performed interestingly, sending an average of 50 messages, but receiving 10 times as much. This is likely due to many receipts that would be received during the interval when the radio is turned on before transmitting a message occurs. These messages are from other data collectors nearby and multihop requests from other clusterheads. Data collectors are also programmed to sleep at intervals, saving resources.

be improved. The environmental models for obstacles, requires further work for radio attenuation factors. Additionally, there needs to be a method of managing the size of the belief-base used in each agent, as this turns out to grow unboundedly in a case such as flooding. Finally, the simulator could be improved by providing a better interface and a model for radio channels.

VII. Conclusions

This work has discussed the potential for using autonomous agents in wireless sensor networks. In particular, it has outlined the vision for such a network, and the work involved to date in using agent mechanisms in order to improve current WSN performance, flexibility, scalability, and adaptiveness. Since deploying such systems is difficult to debug, early testing is important. This work has presented a partial solution; a simulator that is both general purpose, and that also incorporates the work of deliberative agent research. Above all, the simulations have shown that the strong agents used in the simulation can model wireless sensor network behaviors. However, the benefits gained from such agency remains to be explored in future work.

VIII. Acknowledgments

This work has been supported by the ArsLogica research group. Thanks to Luca Debiasi, Fabrizio Stefani, and Fernando Pianegiani, for their interest, feedback, and time spent in discussions and demos of the simulator.

References

[1] ArsLogica, STATUS Project. http://www.arslogica.it/projects/status/status.html, Aug. 2008.

[2] Samson: An agent oriented wireless sensor network simulator. http://lama.disi.unitn.it/page.php?34, Nov. 2008.

[3] K. Akkaya and M. F. Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3(3):325–349, 2005.

[4] J. Beutel. Metrics for sensor network platforms. In *REALWSN 06*. ACM, 2006.

[5] R. H. Bordini, M. Wooldridge, and J. F. Hübner. *Programming Multi-Agent Systems in AgentSpeak using Jason (Wiley Series in Agent Technology)*. John Wiley & Sons, 2007.

[6] L. Chrisman, R. Caruana, and W. Carriker. Intelligent agent design issues: Internal agent state and incomplete perception. In *In AAAI Fall Symposium Series: Sensory Aspects of Robotic Intelligence*, pages 18–25. Press/MIT Press, 1991.

[7] R. W. Collier. *Agent Factory: A Framework for the Engineering of Agent-Oriented Applications*. Doctor of philosophy, National University of Ireland, Faculty of Science, University College Dublin, December 2002.

[8] Crossbow. Crossbow sensors. http://www.xbow.com/index.aspx, Aug. 2008.

[9] C.-L. Fok, G.-C. Roman, and C. Lu. Mobile agent middleware for sensor networks: an application case study. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, page 51, Piscataway, NJ, USA, 2005. IEEE Press.

[10] N. R. Jennings. On agent-based software engineering. *Artificial Intelligence*, 177(2):277–296, 2000.

[11] H. Karl and A. Willig. *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, 2005.

[12] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[13] J. Kho, A. Rogers, and N. R. Jennings. Decentralised adaptive sampling of wireless sensor networks. In *in 1st Int Workshop on Agent Technology for Sensor Networks*, 2007.

[14] P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim: accurate and scalable simulation of entire tinyos applications. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 126–137, New York, NY, USA, 2003. ACM.

[15] R. Ma, G. M. P. O'Hare, and M. J. O'Grady. Embedded intelligence: Enabling in-situ power management for wireless sensor networks. In *EuroSSC*, pages 244–247, 2006.

[16] A. S. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. In R. van Hoe, editor, *Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Eindhoven, The Netherlands, 1996.

[17] A. S. Rao and M. P. Georgeff. BDI-agents: from theory to practice. In *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco, 1995.

[18] G. F. Riley. The georgia tech network simulator. In *MoMeTools '03: Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, pages 5–12, New York, NY, USA, 2003. ACM.

[19] A. Rogers, E. David, and N. Jennings. Self-organized routing for wireless microsensor networks. *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 35(3):349–359, May 2005.

[20] L. Ruiz, T. Braga, F. Silva, H. Assuncao, J. Nogueira, and A. Loureiro. On the design of a self-managed wireless sensor network. *Communications Magazine, IEEE*, 43(8):95–102, Aug. 2005.

[21] Sentilla. Sentilla.com. http://www.sentilla.com/index.html, Aug. 2008.

[22] Sentilla. Tmote sky datasheet. http://www.sentilla.com/pdf/eol/tmote-sky-datasheet.pdf, Aug. 2008.

[23] A. Sobeih, W.-P. Chen, J. C. Hou, L.-C. Kung, N. Li, H. Lim, H.-Y. Tyan, and H. Zhang. J-sim: A simulation environment for wireless sensor networks. In *ANSS '05: Proceedings of the 38th annual Symposium on Simulation*, pages 175–187, Washington, DC, USA, 2005. IEEE Computer Society.

[24] M. Vinyals, J. A. Rodrguez-Aguilar, and J. Cerquides. A survey on sensor networks from a multi-agent perspective. In *2th International Workshop on Agent Technology for Sensor Networks (ATSN-08)*, 2008.

[25] T. D. Wolf and T. Holvoet. Towards autonomic computing: agent-based modelling, dynamical systems analysis, and decentralised control. In *In Proceedings of the First International Workshop on Autonomic Computing Principles and Architectures*, page 10, 2003.

[26] O. Yadgar and S. Kraus. Coverage density as a dominant property of large-scale sensor networks. In *CIA*, pages 138–152, 2006.

[27] F. Zboril and V. F. Zboril. Simulation of wireless sensor networks with intelligent nodes. In *10th International Conference on Computer Modelling and Simulation*, page 6. IEEE Computer Society, 2008.