Title	Realising an Agent-oriented middleware for Heterogeneous Sensor Networks
Author(s)	Muldoon, Conor; Tynan, Richard; O'Grady, Michael J.; O'Hare, G. M. P. (Greg M. P.)
Publication date	2008
Publication information	Issarny, V. and Schantz, R. (eds.). Middleware 2008 : ACM/IFIP/USENIX 9th International Middleware Conference, Proceedings
Conference details	Paper presented at Middlware 2008 ACM/IFIP/USENIX 9th International Conference on Middleware, Leuven, Belgium, December 1st-5th, 2008
Publisher	ACM
Link to online version	http://doi.acm.org/10.1145/1462735.1462756
Item record/more information	http://hdl.handle.net/10197/1335
Publisher's version (DOI)	http://dx.doi.org/10.1145/1462735.1462756

Muldoon, C., Tynan, R., O'Grady, M. J., and O'Hare, G. M. P. Realizing an agent-oriented middleware for heterogeneous sensor networks. In Proceedings of the ACM/IFIP/USENIX international Middleware Conference on Middleware, (Leuven, Belgium, December 01 - 05, 2008). ACM, New York, NY, 82-83.

Realizing an Agent-oriented Middleware for Heterogeneous Sensor Networks

Conor Muldoon, Richard Tynan, M.J. O'Grady, G.M.P. O'Hare

CLARITY: The Centre of Sensor Web Technologies
School of Computer Science & Informatics
University College Dublin (UCD)
Belfield, Dublin 4, Ireland.
{conor.muldoon, richard.tynan, michael.j.ogrady, gregory.ohare}@ucd.ie

ABSTRACT

Classic computing systems are characterised by heterogeneity, with its inherent advantages and disadvantages. This raises a number of difficulties for software engineers. The vision offered by a mix-and-match approach is an attractive one, though its practical realisation comes at a cost, as the process of integration is rarely smooth. This scenario will be repeated as Wireless Sensor Networks are increasingly incorporated into mainstream computing. One potential paradigm for managing this heterogeneity is that of intelligent agents. This paper considers the viability and potential of lightweight agents as a paradigm for harnessing the potential of heterogeneous wireless sensor networks.

Categories and Subject Descriptors

C2.1 [**Computer Communications Networks**]: Distributed Systems—*Distributed applications*

General Terms

Design

Keywords

Intelligent Agents, Wireless Sensor Networks

1. INTRODUCTION

There is an obvious attraction in incorporating select prefabricated software components to fashion new applications and services. Nonetheless, integrating diverse components to deliver a coherent system is frequently problematic due to a lack of compatibility. Standardization has been promoted as a solution and it must be acknowledged that the development of good standards and a strict adherence to them significantly reduces and may eliminate many incompatibility issues. Despite the best efforts of standardisation organisations, however, incompatibilities continue to plague conventional computing systems, and will affect Wireless Sensor Networks (WSNs) to some degree. Thus a key challenge facing developers concerns how WSNs may be integrated seamlessly with existing software architectures.

There is a tendency to regard WSNs as being dedicated to singular objectives, such as temperature measurement and so on. In specific application domains, this view is understandable. When WSNs are deployed with commercial objectives in mind, however, they will necessitate multitasking; indeed, over the lifespan of the network, the application will be augmented with capacities to perform additional functions. Thus, heterogeneous WSNs may become the de facto deployment WSN configuration, as those responsible for the network will try to maximise the services supported, power and performance costs notwithstanding. In practice, a WSN will comprise a suite of sensors differing in a multitude of ways. Implementing a software solution to manage this diversity and complexity is a formidable challenge, as is its subsequent integration into legacy systems. While a number of potential software approaches exist, the discussion here focuses on the possibilities offered by the intelligent agent paradigm.

Middleware '08 Companion, December 1-5, 2008 Leuven, Belgium.

2. EMBEDDED AGENTS

Embedded agents enable the extension of Multi-Agent Systems (MASs) to encompass a multitude of low-powered computationally challenged devices. Their potential has been documented elsewhere [2] [6]. Such agents have significant potential in WSN applications either for managing clusters of nodes or for facilitating interoperability with legacy systems. Their versatility has motivated a number of researchers to extend documented MAS frameworks to operate on embedded devices, for example 3APL-M [3]. In the case of WSNs, Agilla [1] represents a middleware platform that has harnessed the agent paradigm. For the purposes of this implementation, Agent Factory Micro Edition (AFME) [5] [4] has been harnessed.

2.1 AFME

AFME is a minimised footprint intelligent agent platform that has been designed for use with resource constrained mobile devices, such as mobile phones and high specification sensor motes, for example, Sun SPOTs. In AFME,

Muldoon, C., Tynan, R., O'Grady, M. J., and O'Hare, G. M. P. Realizing an agent-oriented middleware for heterogeneous sensor networks. In Proceedings of the ACM/IFIP/USENIX international Middleware Conference on Middleware, (Leuven, Belgium, December 01 - 05, 2008). ACM, New York, NY, 82-83.

agents perceive their environment through software components known as perceptors. Perceptors contain imperative functionality for reading hardware sensor values. Agents build a model of the world, their belief set, based upon perceived data. A declarative programming language, based on a logical formalism of belief and commitment, is used to specify the conditions under which commitments should be adopted. The following is an example of an AFME commitment rule:

```
message(request, ?sender, data), informTime(?t),
data(?d) > ?sender, ?t, true, inform(?sender, ?d);
```

In the above rule, if the agent receives a request for data, it adopts a commitment to send the data to the agent that sent the message at time *?t*. The first order structures to the left of the implication (the > symbol) represent beliefs; those to the right represent arguments to the commitment. Arguments preceded by the *?* character represent variables.

A commitment represents an intended course of action. Commitments are useful because they have a stabilising effect on system behaviour. Consider the situation in which an agent is situated in a highly dynamic environment and an event occurs. What should an agent do? replan or continue operating? If the agent always replans, the system will become unstable; but if it never replans, the system will not be adaptive. In AFME, agents commit to actions, but they revise their commitments at various points throughout execution. If a commitment is no longer relevant, it is dropped. When a commitment is adopted, it represents either a primitive action or a plan. Plans are ultimately executed as a series of primitive actions. When a primitive action is executed, an actuator is fired. An actuator provides the imperative code for affecting the environment or system state; there are actuators for sending messages over the network and calibrating sensors, for instance.

3. IMPLEMENTATION

In this prototype, the migration and execution of agents across an array of heterogeneous sensing and computing environments is supported. Agent migration refers to the process of transferring an agent from one device to another with its data intact so that it can continue operating at its destination. Migration is achieved through data duplication. An agent's state is transferred to its target and then deleted at the source. Migration has a number of advantages. For instance, it can be more efficient for a group of agents to migrate to a single device and communicate locally, rather than communicating remotely. For this implementation, agents can migrate between a mobile phone, a Crossbow Stargate, and Sun SPOT motes. Using the Stargate, agents can access information coming from motes. A user interface on the phone provides visualisation of the network state and debugging functionality for the agents.

In AFME, actuators and perceptors interact with platform services, which provide access to hardware components. Actuators and perceptors send and receive information from services through the use of first order structures, which provided an abstract representation of information content. This provides AFME agents with a higher degree of flexibility than object polymorphism in that agents do not obtain direct object references to services and therefore do not have dependencies on the services' class structure. When an agent moves from one sensor to another, it can interact with a different set of services without having to modify the imperative code.

By decoupling the actuators, perceptors, and services, an advantage accrues that enables agents to interact with heterogeneous sensors. The imperative code for calibrating and obtaining information from the sensors on each platform is different and is implemented within local platform services. When an agent is operating on the Stargate, it uses the CDC Java platform and a service that remotely configures Berkeley motes. On Sun SPOTs, the CLDC Java platform is used, and sensors are calibrated locally. Agents migrate from platform to platform, obtaining information from services, and calibrating the various sensors accordingly. Since the agents do not obtain direct object references to the local services as they are operating on them, the developer need only write a single set of actuators and perceptors. In the CLDC Sun SPOT JVM, a type of code migration is possible through the use of isolates. This is not supported in AFME, as it is concerned with agent migration rather than the migration of the entire platform.

4. CONCLUSION

As the range and capabilities of sensors increases, so do the difficulties in developing solutions that facilitate their effective utilisation and incorporation into legacy systems. In this paper, the agent paradigm was considered as a basis for realizing a middleware solution for heterogeneous sensor networks. In particular, implications for agent design were considered in light of this heterogeneity. Future work involves the identification of heuristics that designers can use when considering the use of agents in WSN scenarios.

5. ACKNOWLEDGMENTS

This work was supported by Science Foundation Ireland (SFI) (Grant No. 07/CE/I1147) and the Irish Research Council for Science & Engineering Technologies (IRCSET).

6. REFERENCES

[1] C.-L. Fok, G.-C. Roman, and C. Lu. Mobile agent middleware for sensor networks: An application case study. In *Proc. of the 4th Int. Conf. on Information Processing in Sensor Networks (IPSN'05)*, pages 382–387. IEEE, April 2005.

[2] H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish, and H. Duman. Creating an ambient-intelligence environment using embedded agents. *IEEE Intelligent Systems*, 19(6):12–20, 2004.

[3] F. Koch, J.-J. Meyer, F. Dignum, and I. Rahwan. Programming Deliberative Agents for Mobile Services: the 3APL-M Platform. *AAMAS'05 Workshop on Programming Multi-Agent Systems (ProMAS05)*, 2005.

[4] C. Muldoon. *An Agent Framework for Ubiquitous Services*. PhD thesis, School of Computer Science and Informatics, Dublin, Ireland, 2007.

[5] C. Muldoon, G. M. P. O'Hare, R. W. Collier, and M. J. O'Grady. Agent factory micro edition: A framework for ambient applications. In *International Conference on Computational Science (3)*, pages 727–734, 2006.

[6] M. J. O'Grady and G. M. P. O'Hare. Mobile devices and intelligent agents: towards a new generation of applications and services. *Inf. Sci. Inf. Comput. Sci.*, 171(4):335–353, 2005.