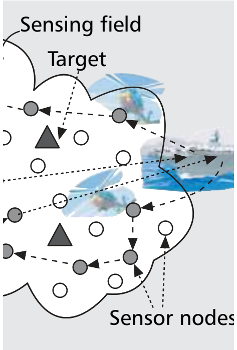
Applications and Design Issues for Mobile Agents in Wireless Sensor Networks

Min Chen, Sergio Gonzalez, and Victor C. M. Leung, University of British Columbia



The authors survey the potential applications of mobile agents in WSNs and discuss the key design issues for such applications. They decompose the agent design functionality into four components, that is, architecture, itinerary planning, middleware system design, and agent cooperation.

Abstract

Recently, research interest has increased in the design, development, and deployment of mobile agent systems for high-level inference and surveillance in a wireless sensor network (WSN). Mobile agent systems employ migrating codes to facilitate flexible application re-tasking, local processing, and collaborative signal and information processing. This provides extra flexibility, as well as new capabilities to WSNs in contrast to the conventional WSN operations based on the client-server computing model. In this article we survey the potential applications of mobile agents in WSNs and discuss the key design issues for such applications. We decompose the agent design functionality into four components, that is, architecture, itinerary planning, middleware system design, and agent cooperation. This taxonomy covers low-level to high-level design issues and facilitates the creation of a component-based and efficient mobile agent system for a wide range of applications. With a different realization for each design component, it is expected that flexible trade-offs (e.g., between energy and delay) can be achieved according to specific application requirements.

Introduction

Wireless sensor networks (WSNs) have attracted much attention in the research community over the last few years, driven by a wealth of theoretical and practical challenges and an increasing number of practical civilian applications. "One deployment, multiple applications" is an emerging trend in the development of WSNs, due to the high cost of deploying hundreds and thousands of sensor nodes over a wide geographical area and the application-specific nature of tasking a WSN [1]. Such a trend requires sensor nodes to have various capabilities to handle multiple applications. However, it is infeasible to store the programs required to run every possible application in the local memory of embedded sensors, as these devices generally have tight memory constraints. The use of mobile agents to dynamically deploy new applications in WSNs is proving to be an effective method to address this challenge. A mobile agent is a special kind of software or computer program that migrates between the nodes of a network to perform a task (or tasks) autonomously and intelligently, in response to changing conditions in the network environment, to realize the objectives of the agent dispatcher. Mobile agents have been found to be particularly useful in facilitating efficient data fusion and dissemination in WSNs [1–10].

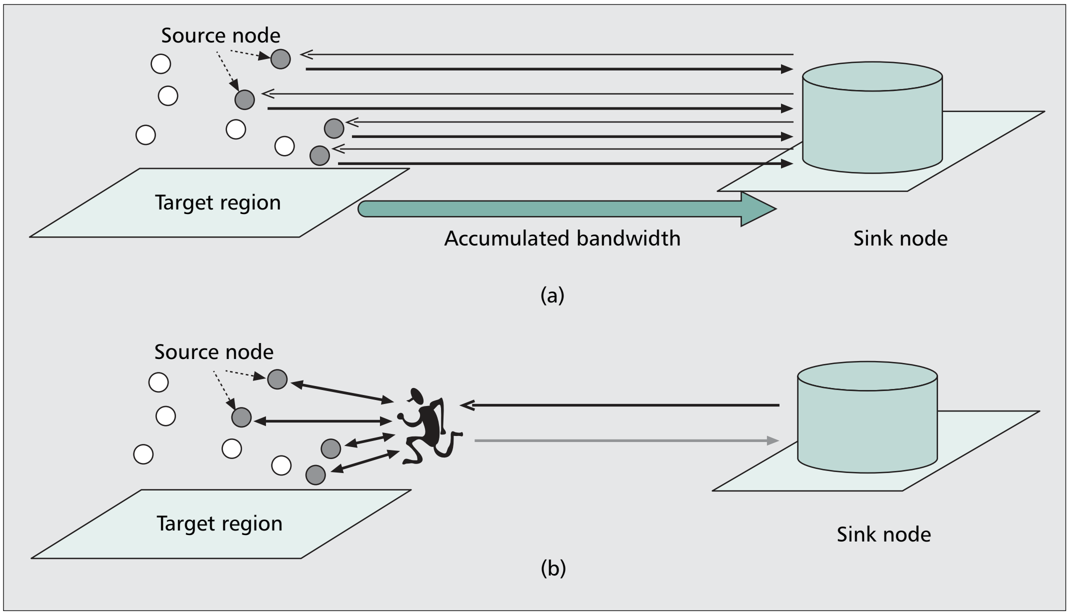
Figure 1a shows the traditional data dissemination method that follows the client-server based paradigm, where the occurrences of certain events trigger surrounding source nodes to collect and send data to the sink individually. In the client-server paradigm, the number of data flows is generally equal to the number of the source nodes, leading to high bandwidth and energy consumption. Furthermore, this approach could lead to unbalanced energy consumption in the network, due to the fact that nodes closer to the sink forward more data on behalf of other nodes. In the mobile agent-based approach illustrated in Fig. 1b, the sink node sends a mobile agent to the target region to visit the source nodes one by one. The sensed data is reduced and aggregated by the agent and then sent back to the sink as instructed by the mobile agent, yielding a single traffic flow instead of multiple ones. The motivation for using mobile agents in WSNs has been extensively studied in [5]. However, there are also some disadvantages of using mobile agents in particular scenarios, such as code caching and with regard to security issues.

In this article, we provide a survey of the main applications of mobile agents in WSNs and of the research issues in their development. We describe applications and architectures of mobile agent-based data dissemination in WSNs. Mobile agent itinerary planning approaches are described. We discuss the design of mobile agent middleware systems and corresponding mobile agent cooperation. We then conclude the article.

Applications of Mobile Agents

The benefits of applying mobile agent systems in WSNs are mainly twofold. First, they can potentially reduce bandwidth consumption by moving the data processing elements to the location of the sensed data, whose transmissions in the raw otherwise would incur most of the energy expen-

1536-1284/07/$20.00 © 2007 IEEE

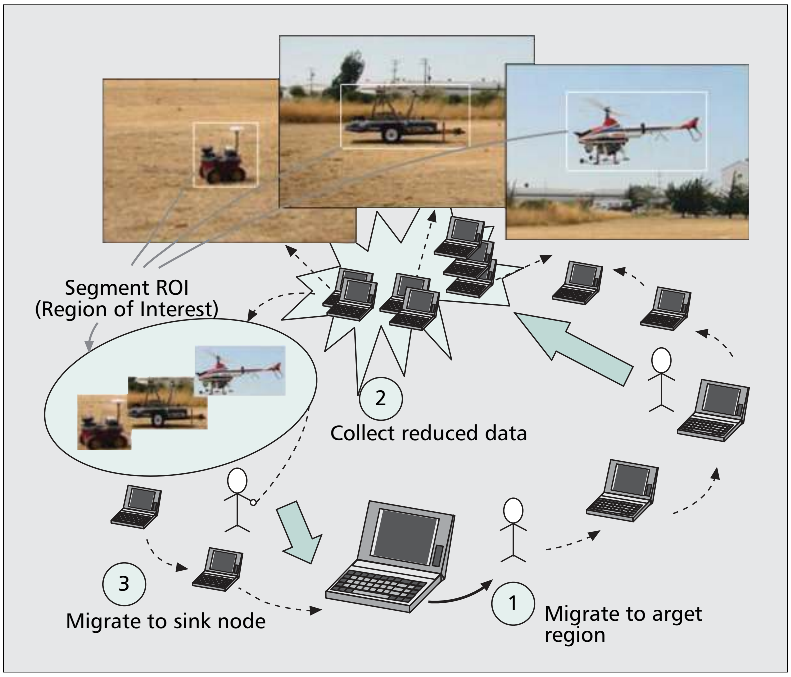


■ **Figure 1.** *Data dissemination in WSNs: a) client-server-based paradigm; b) mobile-agent-based paradigm.*

ditures of the nodes. This is highly appealing when large amounts of data have been collected and must be disseminated to the sink. In addition, mobile agent systems introduce a higher degree of WSN re-tasking flexibility, compared to other approaches, and facilitate collaborative information processing. In light of these aspects, we describe two archetypal WSN applications in which the use of mobile agent technology proves to be an efficient solution. In practice, several other WSN issues, such as routing and data fusion can be effectively tackled by mobile agent systems.

Visual Sensor Network

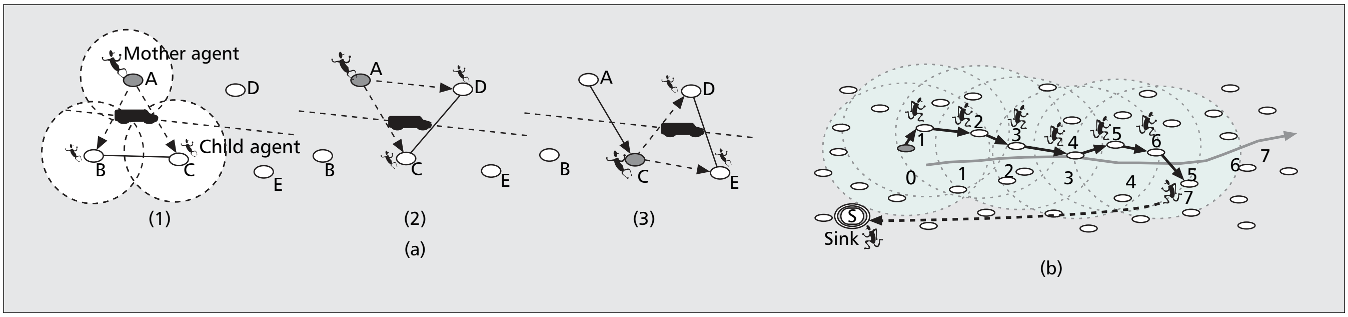
Recent advances in hardware miniaturization have allowed the implementation of sensor devices that support the use of specialized add-on modules for imagery applications. For example, the Cyclops image capturing and inference module can be interfaced with popular WSN devices, such as Crossbow's MICA2 and MICAz for image sensing applications [11]. The availability of such lightweight and inexpensive imaging hardware has fostered the development of visual sensor networks that enable retrieval of video streams and still images from sensor nodes. In fact, mobile agents have been found especially useful in visual sensor networks [1]. Because the amount of data generated by an image sensor is generally very large, transmitting whole pictures not only consumes much bandwidth and energy, but also may be unnecessary if the sink must evaluate only a certain region of interest (ROI) in the picture. Figure 2 illustrates an application of mobile agents in visual sensor networks. Here, a mobile agent that carries image segmentation codes is dispatched to the target region to visit the image sensors one by one, collecting image data from their corresponding ROI. Thus, the large volume of imagery data at each sensor node in the target region is reduced to a much smaller one. When the circumstances surrounding the environment being sensed have changed substantially, new mobile agent(s) that carry different image segmentation algorithm(s) can be dispatched to re-task specific image sensors to keep the image processing code working efficiently.



■ **Figure 2.** *Mobile-agent-based image querying.*

Target Tracking

In [2], mobile agents are employed to track the location of a target and report it to a location server periodically by employing a simple localization algorithm called trilateration. In this approach, a node relies on location measurement information from itself and two of its neighbors to estimate the target location, as illustrated in Fig. 3a (1). Here, the three circles



■ **Figure 3.** *Illustration of two target tracking applications: a) geographical location tracking; b) object recognition.*

indicate possible target positions based on measurements from three mobile agents. One is the mother agent, and the other two are child agents that have been invited by the mother agent to cooperatively position the object. In Fig. 3a (1), the mother agent residing at node A dispatches child agents to nodes B and C to help locate the target. When the target moves away from node B, the received signal level at B will be reduced, and when the signal level falls below a threshold, the child agent at B is revoked, and a new child agent is dispatched to D, as shown in Fig. 3a (2). As the target passes node C, the mother agent itself will lose the tracking, in which case it will migrate to C. All old child agents are revoked, and new child agents will be dispatched by the new mother agent to nodes D and E, as shown in Fig. 3a (3). While Fig. 3a shows the case in which the mother agent cooperates with only two child agents, more child agents can be employed to improve the positioning accuracy.

By comparison, Xu et al. [3] proposed a different application of mobile agents for target tracking. As illustrated in Fig. 3b, after a new target is detected, a mobile agent is dispatched to track the roaming path of the target. When the agent migrates to a sensor node, it gathers data to progressively increase the accuracy of recognizing the object. After the achieved accuracy exceeds a certain threshold and satisfies the requirement of object recognition, the mobile agent terminates the tracking process and returns the collected results to the sink node. Thus, the overhead associated with unnecessary data gathering and agent migration is avoided.

Architecture of Mobile Agent-Based Data Dissemination in WSNs

According to the structure of a WSN, we divide the mobile agent-based paradigm into two categories: the architecture in a hierarchical WSN and the architecture in a flat WSN. In a hierarchical WSN, the roles or capabilities of the nodes are not equal, whereas the roles or capabilities of all nodes are identical in a flat sensor network.

Architecture in Hierarchical Sensor Networks

In a hierarchical WSN (e.g., a clustered WSN), the operation of a mobile agent is simplified. In [4], two hybrid mobile agent-based methods are proposed in clustered WSNs. In an *intra-cluster* method, each cluster head dispatches a mobile agent that visits all of the cluster members one by one to collect and aggregate data. When a mobile agent returns to its corresponding cluster head, it sends the aggregated data back to the processing center (i.e., the sink node). In contrast, an *inter-cluster* method does not involve mobile agent operations within a cluster; instead, a mobile agent migrates among cluster heads and the processing center.

The intra-cluster approach favors the scenario where the number of nodes within a cluster is large, but the number of cluster heads is small. In comparison, the inter-cluster approach is more effective when the number of nodes within a cluster is relatively small.

Setting up and maintaining special structures in a hierarchical WSN may require a lot of control overhead. This limitation can be addressed by a flat WSN structure that may be suitable for a wide range of sensor applications. The following section introduces mobile agent architectures in flat WSNs.

Architecture in Flat Sensor Networks

A mobile agent-based distributed sensor network (MADSN) [5] was the first proposed mobile agent-based paradigm that could be adopted in both hierarchical and flat WSNs. As in the flat WSNs shown in Fig. 4a, the sink (i.e., the processing center in [5]) dispatches a specific number of mobile agents to directly collect data in the target region. In [5], source nodes are assumed to be in proximity to the sink. In a case where source nodes are far away from the sink, each agent carrying the mobile code can migrate a long distance before visiting the first source node. The large overhead of sending the mobile codes from the sink every time is not only unnecessary, but also weakens the benefit of using mobile agents. Thus, we propose a novel architecture called mobile agent-based WSN (MAWSN) [6].

To better illustrate the agent operation in MAWSN, we give the following analogy. As shown in Fig. 4b, only one mobile agent, referred to as the *mother agent*, is dispatched from the sink to the target region. The mother agent is analogous to an aircraft carrier. Initially, the mother agent carries the mobile code to a node within the target region and stays there, similar to an aircraft carrier anchoring in the operational theatre. Then, the mother agent launches a number of child agents to execute specific

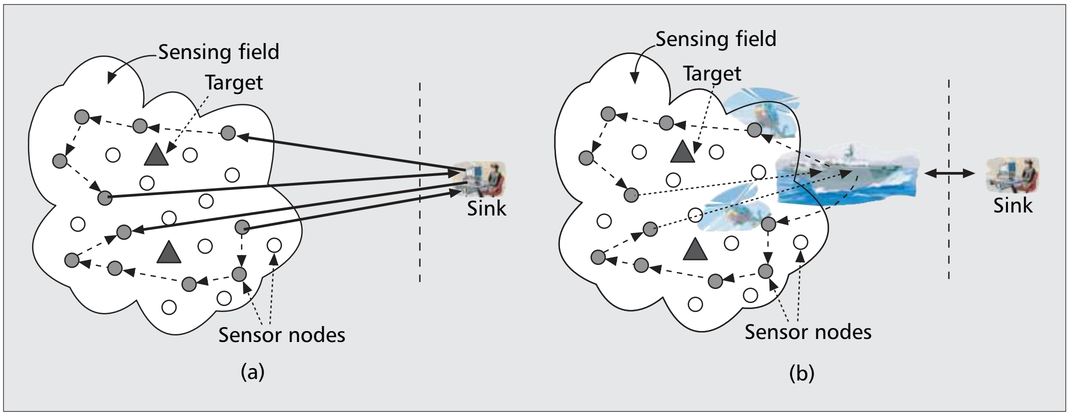


Figure 4. *Mobile-agent-based architecture in flat sensor networks: a) MADSN; b) MAWSN.*

Although global optimization can be achieved using the genetic algorithm, it is not a lightweight solution that is suitable for sensor nodes constrained in energy supply; a way for the source and sink nodes to determine the global network status remains a challenge.

task(s) in sensor nodes within the target region. A child agent, similar to an aircraft going to specific targets, will visit the assigned set of source nodes one by one in a round of data collection and aggregation and carry the collected data either to the mother agent or to the sink directly. To initiate new rounds of data collection, new child agents are launched periodically by the mother agent according to the desired data reporting rate. To reduce latency, multiple child agents also can be launched by the mother agent simultaneously to carry out task(s) in parallel. In MAWSN, the use of mobile agents enables three levels of information redundancy reduction [6] as follows:

- *Node level*: The raw data, generated by individual source nodes, are reduced by child agent-assisted local processing. Then, only relevant information is forwarded to the sink.
- *Child agent level*: It has been shown that when nodes are in close geographical proximity, their measurements display a high degree of correlation. Thus, data aggregation is performed to reduce the redundancy of the sensory data from a single event when the child agent visits source nodes in the vicinity of the event one by one.
- *Mother agent level*: After multiple child agents return to the mother agent, the mother agent can further reduce the redundancy that exists in the collected data by various child agents.

ITINERARY PLANNING

We define an itinerary [3] as the route followed during mobile agent migration. Itinerary planning includes the following two issues that can be addressed by the sink or by the mobile agent autonomously:

- Selection of the set of the source nodes to be visited by the mobile agent
- Determination of a source-visiting sequence in an energy-efficient manner

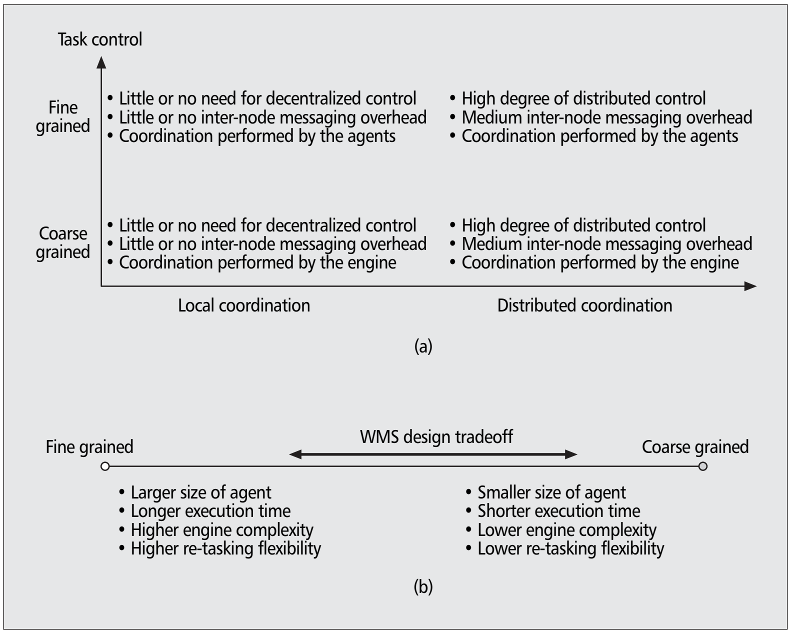
The order in which a mobile agent visits the selected source nodes can have a significant impact on energy consumption. Finding an optimal source-visiting sequence is a non-deterministic polynomial-time (NP)-complete problem. The sequence can be fixed, dynamic, or a combination thereof based on the information of one-hop neighbors and/or the information from previously visited nodes piggybacked by the mobile agent. Itinerary planning can be categorized as:

- Static planning, where the agent itinerary is totally determined by the sink node before the agent is dispatched.
- Dynamic planning, where the mobile agent autonomously determines the source nodes to be visited and the route of migration according to the current network status.
- Hybrid planning, where the set of source nodes to be visited is decided by the sink, and the source-visiting sequence is determined dynamically by the mobile agent.

Static Planning — Static planning makes use of current global network information and derives an efficient agent path at the dispatcher before the mobile agents are sent. Two approaches, local closest first (LCF) and global closest first (GCF), have been proposed [12]. Assuming both algorithms start at the same sensor node closest to the dispatcher, LCF searches for the next node with the shortest distance to the current node, whereas GCF searches for the next node closest to the dispatcher. When source nodes intend to form multiple clusters with similar distance to the sink, GCF causes zigzag routing due to the itinerary fluctuations among those clusters. This limitation can be addressed by LCF, which may be suitable for a wide range of scenarios.

In [7], a genetic algorithm for mobile agent itinerary planning in WSNs is presented. It assumes that each node cannot be visited repeatedly to shorten the search space. Although global optimization can be achieved using the genetic algorithm, it is not a lightweight solution that is suitable for sensor nodes constrained in energy supply; a way for the source and sink nodes to determine the global network status remains a challenge.

Dynamic Planning — Because the global information collected at the sink may become out-dated due to changes over the network as the agent migrates, a static itinerary may become sub-optimal in a dynamic WSN. In contrast, dynamic planning enables the mobile agent to determine the next node to visit at each stop of its migration path. The derivation of a dynamic agent


Task control
Fine grained
• Little or no need for decentralized control
• Little or no inter-node messaging overhead
• Coordination performed by the agents
• High degree of distributed control
• Medium inter-node messaging overhead
• Coordination performed by the agents
Coarse grained
• Little or no need for decentralized control
• Little or no inter-node messaging overhead
• Coordination performed by the engine
• High degree of distributed control
• Medium inter-node messaging overhead
• Coordination performed by the engine
Local coordination
Distributed coordination
(a)
Fine grained
WMS design tradeoff
Coarse grained
• Larger size of agent
• Longer execution time
• Higher engine complexity
• Higher re-tasking flexibility
• Smaller size of agent
• Shorter execution time
• Lower engine complexity
• Lower re-tasking flexibility
(b)

■ **Figure 5.** *WMS programming guideline: a) task control model; b) trade-off map.*

route must take into consideration the trade-offs between the cost of migration and the benefit of the migration accuracy. In [3], a dynamic planning method is proposed to achieve progressive fusion accuracy at a minimal expense of resources. The dynamic planning approach seeks the sensor node that

- Has the largest available energy remaining
- Requires the least energy consumption for the agent's migration
- Provides the greatest amount of information gain

As discussed earlier, the mobile agent always should try to migrate to a sensor node that increases the accuracy progressively so that the number of nodes that the mobile agent must visit to recognize the object is reduced.

Hybrid Planning — In hybrid planning, the selection of the source-visiting set is static, whereas the decision of the source-visiting sequence is dynamic. In [1], a hybrid planning scheme called mobile agent-based directed diffusion (MADD) is proposed. In MADD, if the sources in the target region detect an event of interest, they flood exploratory packets to the sink individually. Based on these exploratory packets, the sink statically selects the sources that will be visited by a mobile agent, which autonomously decides on the source-visiting sequence as it migrates among the nodes in the source-visiting set. As a result, the mobile agent follows a cost-efficient path among target sensors in MADD.

Middleware Layer Design

Mobile agents are often implemented as middleware. Middleware is used widely in computing systems to bridge the gap between the operating system and high-level components and to facilitate the development and deployment of applications. Mobile-agent middleware designed for deployment over resource-rich wireline networks can easily provide enough functionality to perform a wide variety of tasks. However, the severe hardware and energy limitations in WSNs lead to many technical challenges in designing a practical wireless sensor network mobile-agent middleware system (WMS). A WMS provides a platform to support mobile agents to perform application-specific tasks, while enabling rapid application development in WSNs. Trade-offs arise from the requirement to drastically simplify the design due to the resource constraints, while maintaining a degree of functionality that is sufficient to warrant its application in WSNs. SensorWare [8] and Agilla [9] are pioneer frameworks in WMS design that will be reviewed in this section together with more current developments.

One way to achieve the goal of simplifying a WMS while keeping re-tasking flexible is by augmenting the WMS language constructs, especially to make the task control part of the agents more expressive. This approach not only helps to reduce the size of an agent code, but also simplifies the design of the code interpreter itself, while promoting the use of native function libraries. Consequently, programmers implementing WSN applications might rely on preexisting node functionality that the agents can employ in-situ, relieving the agents of low-level chores and reducing the size of the agent code.

However, the previous reasoning might not always apply. For instance, the resources of the WSN nodes might be severely constrained so as to limit the number of native library functions. In these cases, the use of low-level language constructs to enable fine-tuned control of specific tasks by the agents becomes more appealing, at the expense of a more complex code interpreter.

The task control capabilities of a WMS should thus cater to the requirements of the intended application. We can categorize WMS designs as being *coarse-grained*, which employ a high-level of code abstraction for task control, or as *fine-grained*, where the code abstraction is at a low-level. For example, a high-level language construct similar to a tool command language tool kit (TCL/TK) is employed in SensorWare [8], where a single code expression can encompass a number of lower-level tasks. In contrast, Agilla [9] employs mostly low-level code instructions, similar to the assembly code employed to program microprocessors.

Depending on the resource constraints of the WSN nodes, the task control can be geared toward distributed task coordination, or local processing of data, as summarized in Fig. 5a. Fine-grained task control leads to a higher level of program execution flexibility, but also to a potentially longer program, especially if the WMS is designed to work with interpreted code scripts. Evidently, this would lead to a more complex WMS overall, which ultimately has an impact on the node resources. Conversely, a coarse-grained model leads to simple programs (or mobile codes) that instruct the nodes as to what must be done, while promoting the use of native library functions in the nodes that execute faster. As a result, the WMS complexity and the size of a mobile agent can be potentially lowered, at the expense of less flexibility in task control and re-tasking. As seen in Fig. 5b, suitable WMS design trade-offs should attempt to satisfy application specific delay and re-tasking requirements while saving energy consumption as much as possible, depending on the capabilities of the sensor nodes.

The previous arguments demonstrate that a WMS is most suitable for applications where the WSN relies heavily on collaborative information processing, as in [5]. It is also apparent that a WMS must be designed with the WSN resource constraints in mind, while preserving as much functionality as possible to promote programming flexibility. Both SensorWare and Agilla have made great strides in accomplishing this objective by enabling collaborations among agents deployed as mobile scripts, as well as the re-tasking of a WSN as required by the users. However, these are intended for use in distinct hardware platforms: Agilla employs assembly-like language constructs for fine-grained task control over the popular but severely resource-constrained Crossbow motes; SensorWare, with improved coarse-grained task processing capabilities, targets WSNs with nodes richer in hardware resources.

To develop a more powerful agent-processing platform for WSNs, we proposed Wireless SEnsor Mobile AgeNts (WISEMAN) [10] that aims to support coarse-grained task coordination as in SensorWare, but with the ability to function over resource-constrained nodes, as Agilla does. This is accomplished by a novel approach whereby a multi-agent application is realized using language constructs for the mobile codes that include the propagation strategy and collaboration technique at a high-level. Therefore, a coarse-grained approach is provided for the agent task control over the WSN as an aggregate, whereas limited fine-grained control is provided to define the thresholds upon which agents make certain decisions (e.g., when a certain condition has been met, or when a task has been completed). As a result, agent code is further compacted, and task control flexibility is preserved. WISEMAN is a work in progress for which experimental implementations are being built for evaluation.

Multiple Agents Cooperation

Mobile agents can work either as single processing units or as a distributed collection of components that can cooperate to achieve a given task. The requirement to provide the means for agent cooperation is an important consideration in WMS design, because this communications mechanism has been shown to reduce energy consumption in the WSN [2, 8, 9]. In addition, multi-agent cooperation facilitates the implementation of agent codes that perform other WSN-related tasks, such as routing. In fact, early research on the subject showed measurable improvements that were directly attributed to a decoupled agent cooperation mechanism in a network mapping system. In essence, information sharing enables agents to learn what other agents have already learned, enabling a faster task completion time, potential bandwidth savings, and energy conservation from which WSNs can benefit. Consequently, multi-agent cooperation potentially can enable a WSN to perform tasks faster and more efficiently by sharing information between agents.

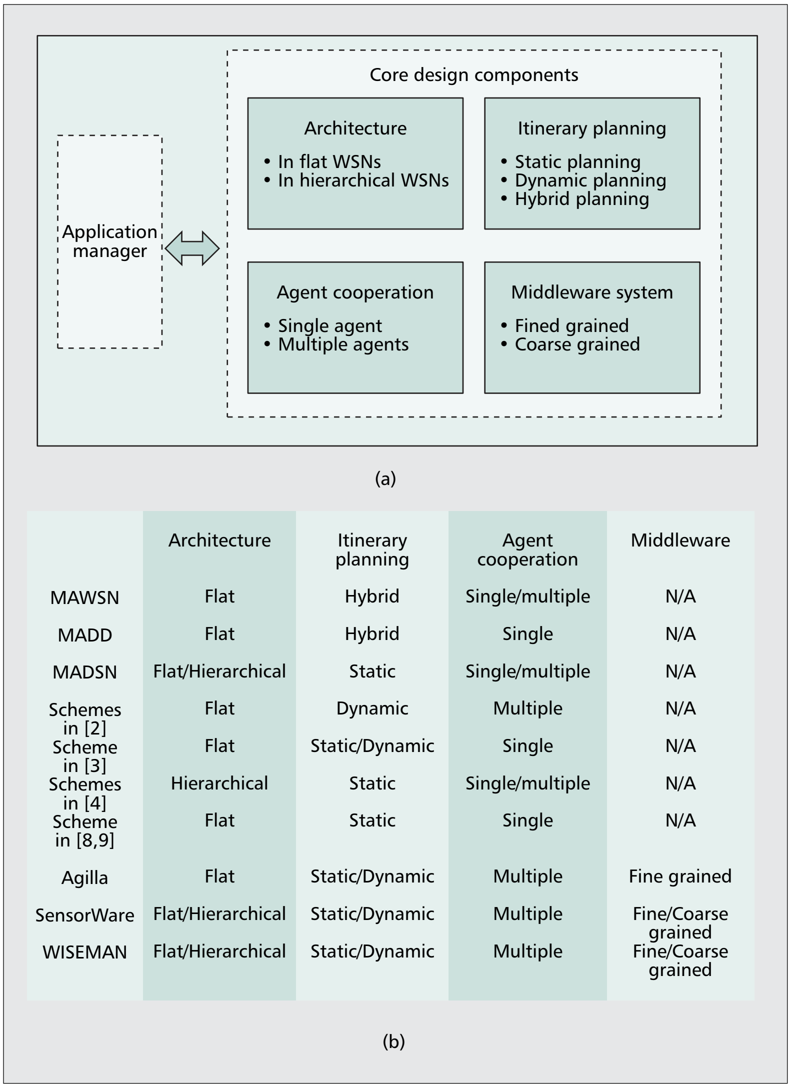
The type of information exchanged by agents might range from simple Boolean values, to complex pre-processed data. This data can be reprocessed later by other agents until the desired result is obtained and can serve as a reference for making decisions in the evolution of the current distributed task. A popular communication mechanism employed by agent systems is the tuple-space, in which typified information is stored at the nodes for other agents to refer to or to modify in a time-decoupled manner. For example, an agent can store a set of three variables representing certain values (e.g., <string agentName, double time, bool foundResult>), whose semantic meaning has been predefined and other agents know how to access. Since data left at a node by an agent might not be accessed by another agent until later, this kind of decoupled information is useful in applications that are not constrained to tight timing requirements; that is, the significance of the data values deteriorates slowly with time, compared to the time granularity of the events being monitored [9].

However, the use of tuple-spaces clearly introduces a certain amount of memory and data processing overhead into the WMS architecture, which calls for a trade-off analysis between system overhead versus performance benefits. On the other hand, direct agent communications might be necessary under either tight timing constraints (as required by real-time applications), or when not doing so would lead to significant use of resources, such as memory or bandwidth. For example, a WSN tasked for image evaluation, as explained previously, might deploy a tightly coupled set of agents searching for evidence of a certain event that was recorded in the form of an image. In this case, agents might directly signal each other with a simple true/false value to indicate whether they were successful in their quest at the current node and as an indication that the search may be required to be extended. In general, WMSs might rely on several agent cooperation mechanisms to accomplish the objectives of the WSN applications.

Conclusion

In general, agent systems can effectively overcome the shortcomings found in other types of problem-solving techniques for WSNs, where bandwidth and power consumption are the main concerns. In this article, we surveyed the major aspects pertaining to mobile agent system deployment over WSNs in terms of applications, architecture, itinerary planning, a middleware system, and multiple agent cooperation. In Fig. 6a, we summarize the core design issues with respect to several mobile agent system components. In Fig. 6b, we categorize the existing work in terms of the core design components. With regard to applications, measurable bandwidth savings can be attained either when large

The requirement to provide the means for agent cooperation is an important consideration in WMS design, because this communications mechanism has been shown to reduce energy consumption in the WSN.



	Architecture	Itinerary planning	Agent cooperation	Middleware
MAWSN	Flat	Hybrid	Single/multiple	N/A
MADD	Flat	Hybrid	Single	N/A
MADSN	Flat/Hierarchical	Static	Single/multiple	N/A
Schemes in [2]	Flat	Dynamic	Multiple	N/A
Scheme in [3]	Flat	Static/Dynamic	Single	N/A
Schemes in [4]	Hierarchical	Static	Single/multiple	N/A
Scheme in [8,9]	Flat	Static	Single	N/A
Agilla	Flat	Static/Dynamic	Multiple	Fine grained
SensorWare	Flat/Hierarchical	Static/Dynamic	Multiple	Fine/Coarse grained
WISEMAN	Flat/Hierarchical	Static/Dynamic	Multiple	Fine/Coarse grained

Figure 6. *Outline for mobile-agent-based system design in WSNs: a) core design components; b) classification and comparison of existing protocols.*

amounts of data are locally processed as instructed by agent codes, or when deployment of a programmable approach that enables task autonomy is required. To this effect, efficient designs for the core components are required to support the scheme being followed by the agent-based application when tackling a particular type of problem. Similarly, the WSN application has a direct influence on the type of communications mechanism employed by the mobile agent system to perform its task efficiently. However, their applicability mainly is warranted not only by the overall energy savings they introduce, but also by the extra flexibility they offer when coping with frequent and/or unexpected aspects of the event being sensed that other types of approaches are unable to address efficiently.

Acknowledgements

This work was supported in part by the Canadian Natural Sciences and Engineering Research Council under grant STPGP 322208-05.

References

[1] M. Chen *et al.*, “Mobile Agent-Based Directed Diffusion in Wireless Sensor Networks,” *EURASIP J. Advances in Sig. Processing*, vol. 2007.

[2] Y. Tseng *et al.*, “Location Tracking in a Wireless Sensor Network by Mobile Agents and Its Data Fusion Strategies,” *Comp. J.*, vol. 47, no. 4, July 2004, pp. 448–60.

[3] Y. Xu and H. Qi, “Mobile Agent Migration Modeling and Design for Target Tracking in Wireless Sensor Networks,” *Ad Hoc Networks*, vol. 6, no. 1, Jan. 2007, pp. 1–16.

[4] Y. Xu and H. Qi, “Distributed Computing Paradigms for Collaborative Signal and Information Processing in Sensor Networks,” *Int’l. J. Parallel and Distrib. Comp.*, vol. 64, no. 8, Aug. 2004, pp. 945–59.

[5] H. Qi, Y. Xu, and X. Wang, “Mobile-Agent-Based Collaborative Signal and Information Processing in Sensor Networks,” *Proc. IEEE*, vol. 91, no. 8, Aug. 2003, pp. 1172–83.

[6] M. Chen *et al.*, “Mobile Agent Based Wireless Sensor Networks,” *J. Comp.*, vol. 1, no. 1, Apr. 2006, pp. 14–21.

[7] Q. Wu *et al.*, “On Computing Mobile Agent Routes for Data Fusion in Distributed Sensor Networks,” *IEEE Trans. Knowledge and Data Eng.*, vol. 16, no. 6, June 2004, pp. 740–53.

[8] A. Boulis, C. Han, and M. Srivastava, “Design and Implementation of a Framework for Efficient and Programmable Sensor Networks,” *Proc. ACM MobiSys ’03*, May 2003, pp. 187–200.

[9] C. Fok, G. Roman, and C. Lu, “Mobile Agent Middleware for Sensor Networks: An Application Case Study,” *Proc. IEEE IPSN ’05*, Apr. 2005.

[10] S. Gonzalez, S. Vuong, and V. C. M. Leung, “A Mobile Code Platform for Distributed Task Control in Wireless Sensor Networks,” *Proc. ACM MobiDE*, June 2006, pp. 83–86.

[11] I. F. Akyildiz, T. Melodia, and K. Chowdhury, “A Survey on Wireless Multimedia Sensor Networks,” *Comp. Networks*, vol. 51, no. 4, Mar. 2007, pp. 921–60.

[12] H. Qi and F. Wang, “Optimal Itinerary Analysis for Mobile Agents in Ad Hoc Wireless Sensor Networks,” *Proc. IEEE ICC ’01*, July 2001, pp. 147–53.

Biographies

MIN CHEN (minchen@ece.ubc.ca) received a Ph.D. degree in electronic engineering from South China University of Technology in 2004. Since 2006 he has been a postdoctoral fellow in the Department of Electrical and Computer Engineering, University of British Columbia, Canada. He was also a postdoctoral fellow in the School of Computer Science and Engineering, Seoul National University.

SERGIO GONZALEZ-VALENZUELA (sergiog@ece.ubc.ca) received an M.A.Sc. degree in data communications from the University of British Columbia in 2002. From 1995 to 1999 he held various engineering positions in Mexico. He is currently nearing the completion of his Ph.D. degree in the Department of Electrical and Computer Engineering, University of British Columbia.

VICTOR C. M. LEUNG [F] (vleung@ece.ubc.ca) received his B.A.Sc. and Ph.D. degrees, both in electrical engineering, from the University of British Columbia in 1977 and 1981, respectively. He is a professor in the Department of Electrical and Computer Engineering, University of British Columbia. He is an editor of *IEEE Transactions on Wireless Communications*, and an associate editor of *IEEE Transactions on Computers* and *IEEE Transactions on Vehicular Technology*.