An Evaluation Framework for middleware approaches on Wireless Sensor Networks

Shuai Tong
Helsinki University of Technology
tong@cc.hut.fi

Abstract

This paper presents an evaluation framework for middleware designs on wireless sensor networks (WSNs). There is lots of existing middleware solutions for WSNs; we introduced middleware approaches and classified them into logical categories according to mechanisms. Based on our evaluation criteria, existing middleware solutions are compared and analyzed in this paper. The evaluation result shows experiences and ideas about middleware solutions regarding to different usages, and also present open issues for further research and study about middleware solutions in WSN.

KEYWORDS: middleware, WSN, sensor network, evaluation framework

1 Introduction

Wireless sensor networks consist of a lot of sensor nodes, which are low cost, small size and low power supply. This kind of networks has some advantages than traditional network, such as easy to deploy, wide scalability, mobility, easy to use in different complicated environments for some special purpose[19]. For instance, WSNs can be widely use for environment monitoring, structure health monitoring, object tracking system, industrial process control and so on[1, 20]. Due to the lack of structure and resource, WSNs also have some limitation compared to traditional network. For example, complicated structure topology when sensor nodes are added or removed; overhead communications needed between nodes and gateway; limitation of node physical resources. Middleware solutions are developed to solve those problems in application. There are lots of existing middleware solutions with different features and mechanisms. We will explain briefly middleware advantages and weaknesses, and reason of why middleware and WSNs suitable to each other. After that we will introduce exist middleware solutions, and classified them into categories, make a comparison and analyze. Finally we give some suggestions about middleware models in different WSN environments and make a table to present conclusion[5].

Middleware solutions are developed as the link between application and low level operating system to solve many wireless sensor network issues, and enhanced application development[8]. It also can act as a layer between sensor nodes and user side server. The middleware support some work phases of WSNs, such as development, maintenance, deployment and data execution[17]. The different types of middleware solutions can provide one or more additional features related to WSNs. For example, abilities of power saving, openness, scalability, mobility and heterogeneity. Those features can improve the performance of WSNs, and solve some practical problems of WSNs[11, 10]. In this paper, we will evaluate and try to march the middleware solutions in various WSNs.

2 An Overview of Existing Middleware on WSN

There are many research groups present a lot of middleware approaches both on theoretical ideas and practical projects. They mainly can be classified into 7 classes according to middleware architectures and approach mechanisms[8], we list them below.

1. Distributed database

   This kind of middleware approaches treats the whole sensor network as a distributed database. Usually it has a friendly and easy to use interface, using SQL-like queries to collect target data. It is good at regularly queries, but lacks to support real time applications, so sometimes it only provides approximate results.

2. Mobile agents

   The main feature of mobile agent middleware is the applications are treat as modules for injection and distribution through the network using mobile codes. The sensor network can implements tasks by transmitting application modules, and transmitting modules could consumes less power than transmitting big applications. It is efficient approach to support multi purpose WSNs and update dynamic applications.

3. Virtual machine

   Virtual machine middleware approach is used to decrease overall power and resource consumption. The system consists of virtual machines and interpreters. Developers write applications into small modules, and injecting and distributing modules through network, and finally virtual machines interpret the modules to implement application.

4. Application driven

For some specified purpose, application driven middleware could adjusts network configurations according to application requirements. It has a structure that supplies multiple network configurations by choosing suitable protocol in its network protocol stack. Different sensors combinations and network configurations provide different performance of QoS to meet related application requirements

5. Message-oriented

   Message-oriented middleware implements the communications using publish/subcribe mechanism between node applications to user side. The publish/subcribe service in the middleware is used to exchange messages between message sender and related receiver, it reduced unnecessary communications when applications collecting data based on specified purpose and events.

6. Component-based

   Component-based middleware implements applications by binding components. By using middleware API, the cross-platform components can be used in the different platform environments, and it allows applications run on various types of devices.

7. Macroprogramming

   Macroprogramming middlewares treat the sensor network as a whole, rather then writing low level software to drive individual nodes. The network behavior is programmed at a high-level specification, and generating node behaviors automatically[9].

For each of category, we will give some existing middleware solutions, and describe theirs mechanisms and features

2.1 Distributed database

- TinyDB

  TinyDB is designed and implemented as Acquisitional Query Processing (ACQP) system for collecting data in a sensor network. Comparing to traditional technology, it is featured such as low power consumption and accurate query results, those are important advantages in the resource limited network environment. TinyDB is a distributed system, it runs on the top of TinyOS operating system, with SQL like interface to execute data from sensor nodes[15].

  For example:

  SELECT nodeid,temp

  FROM sensors

  SAMPLE PERIOD 60s FOR 3600s

  It required reporting collected temperature data from each node in every minute for one hour. The database table in TinyDB is called sensors, it is an unbounded, continuous data stream of values.[15] User could acquire target data by inputting queries from sensors table.[15]

- Cougar

  Cougar is another middleware applying database pattem in sensor network. In Cougar system, there are two types of data: stored data and sensor data. Signal processing functions in each sensor node generate the sensor data, and date are communicated or stored in local as relations in database system[3]. Signal processing functions are modeled by using Abstract Data Type in Cougar. In object-relational database, an ADT represents a same type of sensors in the real world. Cougar also uses SQL like language to implement queries.

- DsWare

  Data Service Middleware (DSWare) is a event-based data centric service middleware. It consists of some services components such as data subscription, event detection, data storage, group management, scheduling and data caching. It uses SQL like language for event registration and cancellation. DSWare has good performance for real time event detection and reduction of overhead network communications, and it also improves the response performance[13].

2.2 Mobile agents

- Agilla

  Agilla is mobile agent based middleware run on TinyOS. Application programs are composed of agents, which be transmitted across the nodes. Each node could support multiple autonomous agents and maintains a neighbor list and tuple space. The agents are transmitted to build an application by following neighbor list. With multiple autonomous agents, it allows multiple users from different applications share the network simultaneously. Agilla has good performance on reliability[6]

- Impala

  Impala is a middleware system using modular programming approach. The whole architecture includes two level layers: upper layer contains all the necessary protocols and application programs. The lower layer contains middleware agents such as Application Updater, Application Adapter, and Event Filter. Impala can support multiple different applications which located in upper layer by adapting, updating and event filtering data from lower layer. The Impala is original designed in ZebraNet Project, which focus on wildlife tracking in large area with few communications devices[14]. It has good performance on mobility, lower event processing time and lower application data transmission volume.

2.3 Virtual machine

- Mate

  Mate is a middleware solution with communication-centric virtual machine for sensor network. It runs on the top of TinyOS operating system, and works as a byte code interpreter. Mate has high level user interface, and it allow using large program by breaking the code into

multiple small pieces: called Capsules. Each capsule contains 24 instructions, and length of each instruction is a byte. The program is injected into sensor network faster and easily by using Mate. It also avoid to produce message buffering, do not need large storage, because sending few byte of data to virtual machine have same effect as sending a large program. [12]

- Magnet

  Magnet is another system-level middleware solution for WSNs and ad hoc networks. MagnetOS is developed as a distributed, power-aware, adaptive operating system. It provides a single system image of a unified Java virtual machine for the nodes in WSN. [2] MagnetOS can divides application into small components automatically, and transmitted them to most suitable nodes in the network. The automatic object replacement policies: NetPull and NetCenter can help to save energy and increase system lifecycle.

2.4 Application driven

- MiLAN

  MiLan for WSNs means Middleware Linking Applications and Networks, it provides the solution that specific applications are allowed to affect the performance of entire network. MiLAN contains network protocol stack to configuration and manage network. It use graph based approach to allow application to know how it performs collected data from different combinations of low level components, and how to choose combination of sensors to satisfy its quality of service requirements. MiLAN is originally designed for medical advising and monitoring, so it have good performance on application reliable[16].

2.5 Message-oriented

- Mires

  Mires is middleware solution runs on TinyOS , and it uses publish/subscribe mechanism to exchange data between nodes and sink nodes. It consists of public/subscribe service, routing components and additional services. The key component is the public/subscribe service. it act as middle layer to transfer publish/subscribe message, and further more to establish communications between local nodes to user applications. Each node advertises the data topics available by its sensors, user applications receives the topics and selects desired data topic which need to be monitored, and after that nodes collect target data and publish them back to user side. In this way, it only transmits target data between target nodes and user side, so it reduced amount of data transmission, and save the energy[18].

2.6 Component-based

- Runes

  Runes means Reconfigurable Ubiquitous Networked Embedded Systems. It is components based middleware solution. Runes middleware consists of Components Frameworks (CFs), which have reusable and dynamic deployable architectures. The high customize services and specific applications can be built by using different components. There is a cross-platform components under layer of middleware API, so Runes solution can be deployed on devices with different platform, from PCs to sensor node with Contiki OS[4].

2.7 Macro programming

- Kairos

  Kairos allows programmer to programming whole sensor network, make a high level specification, and dealing with low-level concerns at each node in network. At beginning, developers write a centralized program for whole application, Kairos preprocessor divided it into subprograms and Kairos compiler compiled them into annotated binary codes. After that the binary codes are distributed to sensor nodes. Annotated binary code is node-specific version contains code to control behavior of each single node. A set of nodes work together to express as a macro level program abstraction. Kairos can decide to use loose node synchronization or tight node synchronization base on programmer's purpose[7].

3 Evaluation and Analyze

3.1 Evaluation Framework

We will evaluate the existing middleware solutions mentioned above base on following evaluation criteria: power saving, scalability, mobility, heterogeneity, and usability.[9]

1. Power saving. Due to the limited power resource of the electrical devices and sensors in WSNs, middlewares should have techniques and mechanisms to manage electrical devices and reduce sensor power consumption. For example, sensors are turned off after use by middleware controlling. In general, sensors use sleep mode to implement basic power saving function. Data transmission overhead is another important factor for power saving, because less data transmission through the network would have less energy consumption. So we determined middlewares which have mechanisms to reduce data transmission through the network as strong support on power saving. For example, Magnet and Mires considered as strong support on power saving.

2. Scalability. *"If an application grows, the network should be flexible enough to allow this growth anywhere and anytime without affecting network performance."*[8] When network topology is dynamic, middleware should support sensor nodes maintenance and reconfiguration. For the middlewares which have mechanisms for protocols updating or switching to ensure network performance, we determined them as strong

support on scalability. For the middlewares which have to update routing information in each sensor, and have effect on network performance, we determined them are weak support on scalability. For example, Mate has strong support on scalability, and cougar is weak on scalability.

3. Mobility. It is middleware ability to keep the communications with mobile sensor nodes in WSNs. Connectivity between the mobile nodes is considered as most important factor. It is related with tracking applications in real world. Some middlewares have special routing protocols and algorithms to implement and improve its mobility. We determined the middlewares which has a set of efficiency ad hoc routing protocols are strong support on mobility. For middlewares which have basic mechanisms for mobility, such as Semantic Rooting Tree, multi-hop routing protocol and so on, we determined them are medium support on mobility. For others middleware which do not have efficiency mechanisms for mobility, we determined them are weak support on mobility. For example, Impala and Mate are determined as strong on mobility, TinyDB is determined as medium, and Milan is determined as weak.

4. Heterogeneity. Middlewares should provide interface to meet various types of hardware and network conditions. When we evaluate heterogeneity in this paper, we determined some middlewares whose have techniques to implement cross-platform applications as providing strong support to heterogeneity, because those middlewares are available for a range of different hardwares and devices. For the middlewares which run on single operating system or virtual machine, we determined them as medium supporting. For the middlewares which only designed for certain hardwares, we determined them as weak supporting. For example, Runes is determined as strong support on heterogeneity, virtual machine based middlewares and middlewares run on the top of TinyOS are determined as medium, and Impala is determined as weak.

5. Usability. It related to whether middleware is easy to use or not. Especially for some complicated WSNs with different sensor types, middlewares should provide friendly user interface according to the usability. Database middlewares use query systems, which use high-level language to implement applications, and SQL like interfaces are easy to use. We determined middlewares which use high-level abstractions and implement GUI are strong support on usability. For the middleware which use instruction set and need to have deep understand on structure, we determined them are weak support on usability. For example, TinyDB is considered as strong support on usability, and Kairos is determined as weak support on usability.

Base on the middleware descriptions in previous section, we will make a general evaluation with evaluation criteria, results are shown in table 1. We will mark them with 3 levels: Strong means selected middleware is strongly support on criteria, Medium means partly support on criteria, and Weak means weak support on criteria.

3.2 Analyze and Application Suggestions

The WSN are widely used in 4 kinds of applications nowadays: Surveillance, Objective Tracking, Industrial Controlling and Data collecting in special purpose.

Surveillance tasks need to work continuously during the work process, and usually it spends a long time. It also could uses various types of devices, so surveillance applications need to pay attention to criteria of power saving and usability. Objective tracking tasks have similar requirements with surveillance tasks. Besides, both objectives and tracking nodes could be mobile, and nodes network could be changed, so the middleware features of mobility and scalability are important. Totally, Objective tracking applications need abilities of power saving, scalability and mobility. For industrial controlling purpose, the nodes in network usually are static, and power supply is stable. Industrial controlling applications need abilities of scalability and heterogeneity. For general data collecting, usability and heterogeneity are considered because of the general purpose.

We would like to give some middleware suggestions according to the result of criteria evaluation in previous section. The suggestion results are shown in table 2. If middleware strongly support on all the related criteria of applications, the middleware is determined as strong support on applications. If one or two of criteria of applications is medium support by middleware, but others criteria are strongly supported, the middleware is determined as medium support on applications. If more than two criteria of applications are medium support or at least one of criteria is weak support by middleware, the middleware is determined as weak support on applications.

4 Conclusion

As the active research area of computer science and telecommunications, middleware solutions in wireless sensor network always be discussed in worldwide and more new designs and projects are under research. In this paper, we described a list of typical existing middleware solutions, give a evaluation according to their features and WSNs applications requirements. Our contribution is to make an evaluation for various middleware solutions, and present middleware suggestions for different types of WSN applications. We hope those ideas and experiences care helpful for further middleware designing and researching in the future.

References

[1] L. F. Akyildiz, Y. S. Weilian Su, and E. Cayirci. A survey on sensor networks. 40, 2002.

[2] R. Barr, J. C. Bicket, D. S. Dantas, B. Du, T. D. Kim, B. Zhou, and E. G. Sirer. On the need for system-level support for ad hoc and sensor networks. In *ACM SIGOPS Operating Systems Review*, volume 36, 2002.

Middleware	Power Saving	Scalability	Mobility	Heterogeneity	Usability
Distributed database					
TinyDB	Strong	Medium	Medium	Medium	Strong
Cougar	Medium	Weak	Weak	Weak	Strong
DSWare	Strong	Medium	Weak	Weak	Strong
Mobile agents					
Agilla	Strong	Strong	Medium	Medium	Weak
Impala	Strong	Strong	Strong	Weak	Strong
Virtual machine					
Mate	Strong	Strong	Strong	Medium	Weak
Magnet	Strong	Strong	Strong	Medium	Strong
Application driven					
MiLAN	Medium	Strong	Weak	Weak	Strong
Message-oriented					
Mires	Strong	Strong	Medium	Medium	Medium
Component-based					
Runes	Medium	Strong	Strong	Strong	Medium
Macro-programming					
Kairos	Medium	Medium	Medium	Medium	Medium

Table 1: Middleware features table

[3] P. Bonnet, J. Gehrke, and P. Seshadr. Towards sensor database systems. In *Proceedings of the Second International Conference on Mobile Data Management*.

[4] P. Costa, G. Coulson, C. Mascolo, G. P. Picco, and S. Zachariadis. The runes middleware: A reconfigurable component-based approach to networked embedded systems. Technical report, 2005.

[5] D. Culler, D. Estrin, and M. Srivastava. Overview of wireless sensor networks. *IEEE Computer Special Issue in Sensor Networks*, 8 2004.

[6] C.-L. Fok, G.-C. Roman, and C. Lu. Rapid development and flexible deployment of adaptive wirelesssensor network applications. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, 2005.

[7] S. Hadim and N. Mohamed. Macro-programming wireless sensor networks using kairos. In *International Conference on Distributed Computing in Sensor Networks*, 2005.

[8] S. Hadim and N. Mohamed. Middleware challenges and approaches for wireless sensor networks. 7, March 2006.

[9] S. Hadim and N. Mohamed. Middleware for wireless sensor networks: A survey. In *First International Conference on Communication System Software and Middleware*. IEEE computer Society, 2006.

[10] W. B. Heinzelman, A. L. Murphy, H. S. Carvalho, and M. A. Perillo. Middleware to support sensor network applications. 18, 2004.

[11] K. Henricksen and R. Robinson. A survey of middleware for sensor networks: state-of-the-art and future directions. In *ACM International Conference Proceeding Series*, volume 218, 2005.

[12] P. Levis and D. Culler. Mate: A tiny virtual machine for sensor networks. 2002.

[13] S. Li, Y. Lin, S. H. Son, J. A. Stankovic, and Y. Wei. Event detection services using data service middleware in distributed sensor networks. In *Telecommunication Systems*, volume 26, pages 351–368, 2004.

[14] T. Liu and M. Martonosi. Impala: A middleware system for managing autonomic, parallel sensor systems. In *Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming*. ACM Press.

[15] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tinydb: An acquisitional query processingsystem for sensor networks. 6, 2005.

[16] A. Murphy and W. Heinzelman. Milan: Middleware linking applications and networks. Technical report, 2002.

[17] K. Romer, O. Kasten, and F. Mattern. Middleware challenges for wireless sensor networks. 6, 2002.

[18] E. Souto, G. Guimaraes, G. Vasconcelos, M. Vieira, N. Rosa, C. Ferraz, and J. Kelner. Mires: a publish/subscribe middleware for sensor networks. 10, 2005.

[19] Wikipedia. en.wikipedia.org/wiki/wsn.

Middlewares	Surveillance	Objective Tracking	Industrial Controling	Data collecting
Distributed database				
TinyDB	Strong	Medium	Medium	Medium
Cougar	Medium	Weak	Weak	Weak
DSWare	Strong	Weak	Weak	Weak
Mobile agents				
Agilla	Weak	Medium	Medium	Weak
Impala	Strong	Strong	Weak	Weak
Virtual machine				
Mate		Strong	Medium	Weak
Magnet	Strong	Strong	Medium	Medium
Application driven				
MiLAN	Medium	Weak	Strong	Weak
Message-oriented				
Mires	Medium	Medium	Medium	Medium
Component-based				
Runes	Medium	Medium	Strong	Medium
Macro-programming				
Kairos	Medium	Medium	Medium	Medium

Table 2: Middleware suggestion for WSNs

[20] Y. Yu, B. Krishnamachari, and V. K. Prasanna. Issues in designing middleware for wireless sensor networks. *IEEE Network*, 18:15–21, 2004.