Service Oriented Wireless Sensor Network Toolbox for Consumer Applications

Jongwoo Sung, Taehong Kim, Seonghoon Kim, Kyubaek Kim, Jang Gwan Im, Daeyoung Kim

***Abstract*— Service-oriented computing which provides flexible composition of various applications using multiple reusable services has getting more attractive. We propose a service oriented wireless sensor networks toolbox, which encapsulates complexities of sensor networks and enables simple service compositions using an intuitive GUI. We utilize sensor networks as a collection of services that are discovered and coordinated by users. Service metadata which are self-describing service capabilities and interfaces are defined and exposed via our service metadata repositories. A service oriented sensor network toolbox consisting of sensor networks, an intuitive GUI application that retrieves metadata for discovered services and helps users to composite various roles are presented.**

***Index Terms*—service oriented computing, sensor network, sensor metadata, networked physical world**

I. Introduction

THOUGH wireless sensor networks have been adopted in varieties of application fields, they tend to be considered tools for complex engineering or scientific tasks. Because dealing with wireless sensor networks is not easy and it often requires good knowledge of embedded system, sensors, and networking, they have been less considered for consumer products.

As opposed to established wireless sensor networks which are tightly coupled with specific applications, service-oriented computing paradigm enables flexible composition of various applications using multiple reusable services [1]. Service oriented computing uses services as fundamental elements for developing applications, and basic service operations such as publications, discovery, selection, and bindings are provided in order to help flexible compositions [1].

Since service oriented computing allows flexible software compositions using loosely coupled services with applications, there have been increasing efforts for adopting a service oriented computing paradigm into wireless sensor networks. In a service-oriented WSN application, each activity (sensing, aggregation, service discovery, etc.) is implemented as a separate service [2]. Since these services are open and self-descriptive, various services from different venders can be utilized together. Services may consist of multiple sensor nodes, and they are reusable and shared among multiple applications [3]. Service oriented sensor provides interoperability, scalability, and reusability of sensors [3].

Underlining technology for service oriented computing is self-describing metadata that explains services. Services expose relevant metadata so that applications or users can use them to understand the service capabilities and interfaces. However, very little work has been done on supporting such important metadata effectively in service oriented sensor networks while most service oriented sensor network researches focused on software abstraction layers. Some featured works are a middleware platform for pervasive spaces [4], management architecture [5], and programming frameworks [2]. Although a few other works [6-9] addressed problems of service interfaces, descriptions and web service based architecture, they tended to leverage established service oriented computing technologies such as OSGi [14] and web services for sensor networks.

However, service oriented computing operations such as service metadata publications, discovery, and binding cannot simply apply to sensor networks due to resource limitations of each sensor node. It is inefficient for every sensor node to store its metadata in precise memory or to have software layers to support web services.

Our works are distinguished from previous works in that we design Internet based metadata architecture that stores metadata and allow users to download it. To associate services consisting of one or a group of sensor nodes with service metadata, we adopted a networked physical world approache [10] that uses unique identifiers, information servers, and identifier resolving system. Based on Internet based metadata architecture, specifically, we present a sequence of service composition processes from service discovery, service browsing and event based service planning, to result interpretations.

We paid attention to possibilities of sensor networks used for general users without professional knowledge about wireless sensor networks. Thus, we design and implement a software toolbox consisting of various services which encapsulates complexities of sensor networks and an intuitive GUI to composite applications using services. Discovering available services and exposing their capabilities are supported via our identification mechanism and Internet metadata architecture. Our service oriented approach enables easy utilization of sensor networks for smart home environments. We also show how our service metadata approach is helpful for optimization of sensor network communications.

Manuscript received August 15, 2008. This work was supported by Korea Science & Engineering Foundation through the NRL Program.

Jongwoo Sung, Taehong Kim, Seonghoon Kim, Kyubaek Kim, Im Jang Gwan, Daeyoung Kim are with Information and Communications University, DAEJON, Korea (email: jwsung@icu.ac.kr, damiano@icu.ac.kr, shkim1980@icu.ac.kr , lilin@icu.ac.kr, limg00n@icu. ac.kr, kimd@ icu.ac.kr).

II. The networked physical world

Auto-ID Center that is a predecessor of current Auto-ID Labs envisioned a world in which all electronic devices are networked and every object, whether it is physical or electronic, is electronically tagged with information pertinent to that object [10]. The networked physical world utilizes physical tags to enable all physical objects to act as networked nodes.

Taking advantage of the networked tag or nodes, realization of networked physical world depends on three main components; global identifiers, information servers and identifier resolvers. The information about physical object is stored in networked information servers. A globally unique identifier acts as a pointer to the appropriate object information servers. An identifier resolver called the object name service is used to locate the desired information server with given an identifier.

The paradigm of networked physical world aimed at diverse ubiquitous automated applications, especially for supply chain management. We adopted the idea of networked physical world and extend it for the service oriented sensor networks.

III. Service oriented sensor networks

A. Service discovery

Since services in wireless sensor networks are not permanent or static, but changeable according to sensor node replacements due to exhausted battery, topology changes or movements of sensor nodes. Service discovery allows consumers to know available services at the time of service compositions. Service discovery depends on unique identifiers of servicing sensor nodes. **N**etwork address and MAC address in wireless sensor networks are identifiers for sensor nodes and network interfaces respectively. However, network addresses are basically changed as network topology changes, and MAC address does not give any useful information about the associated service. For such reasons they are not suitable for service discovery.

As a result additional service identifiers for identifying each servicing sensor nodes are needed. They should be application level names so that they are not changed even if configurations or topologies of sensor networks are changed. In addition, to coordinate services from different venders the identifiers need to be globally unique. Electronic product code (EPC), which is defined by EPCglobal [12] is used for leveraging information technologies in a supply chain, is best suited for our realizations for mentioned requirements. EPCs can be considered a kind of application keys to globally identify services and discover the sensor node. Among several EPC formats we adopt 96 bits EPC format which consists of version (8 bits), sensor node manufacturer (28bits), product (24 bits), and sensor node serial number (36 bits).

For the discovery purpose, base station maintains up-to-date service identifier tables, which list global identifiers of discovered services (sensor nodes). Service identifier table consists of service identifiers of discovered sensor nodes, and the table is maintained by our service discovery protocols. To maintain the service identifier list service discovery mechanisms are designed: 1) registration by sensor nodes; 2) update by sensor nodes; and 3) active discovery. Among three methodologies, registration and update are passive while active discovery is actively operated by the base station.

After sensor nodes are turned on, they interchange *hello* messages with base station to check the interoperability. Then, sensor nodes send *registration(identifiers)* message to base station to notify their new existence. The communication negotiation is done in preset time *p* seconds with maximum retries *r*. After time expires or maximum retries *r* failed, the registration will be tried again after default update expire time *q*.

The *update* procedure is same with the one of the registration, but it is issued only if sensor nodes have any change (such as leaving the network) or update time *q* expires. However, there are possibilities which sensor nodes leave the network without update messages, or registration failed with unexpected errors. To actively figure out this situation quickly as soon as possible, base station sends active *discovery(identifiers)* message to sensor networks in periodic time *s* or when user requests.

Since the service identifiers are not adopted for communications in a sensor network, actual communications between sensor nodes and the base station are done using nodes' network address. For this purpose the base station stores identifiers along with corresponding network address and continuously updates them so that users can communicate with sensor nodes using network addresses.

B. Service browsing

Since service identifiers do not give any idea about the sensor nodes or services except for simple identifications, it is required to retrieve information about services for discovered identifiers. Service metadata consist of attributes-values pairs, and they self-describe all aspects of services: sensor nodes, sensors, sensor data, and applications.

Thus service metadata, defined as data about services, gives the idea about functional operations of them. The typical example of sensor network service metadata includes sensor type, sensing unit, measurement range, sensitivity, coefficients, product description, serial number, aggregation capabilities, message format, and calibration information. Sensor metadata is essential for accessing sensor nodes, and understanding sensing data and events; they are used for a number of service utilization operations such as service browsing, service planning, service optimization, and result interpretations.

The service metadata is described in XML for high level interoperability and data centric presentations of metadata, but XML description requires almost 10 times bigger size than description using custom defined encoding [13]. A simplified example of XML format is shown in Table 1 for an illustration purpose.

It is clearly big burden for resource scared sensor nodes to store them and transfer it to a remote base station via error-prone radio channels. To avoid this problem we store service metadata in distributed Internet repository instead of tiny sensor node memory.

The repository, which we call service metadata repository maintains metadata for services and allows users to query them with target service identifiers. Thus inputs to service metadata repository are identifiers and outputs are service metadata for desired service identifiers. Service metadata repository is different with UDDI used in some early service oriented sensor networks works [6-9]. UDDI is used for looking up specific services using UDDI queries, while main goal of service metadata repository is providing users with service metadata using service identifiers. In addition, service metadata repositories may be maintained by sensor nodes manufacturer or service providers.

Resolving mechanism changes service identifiers to location of relevant service metadata repositories. We adopt object name service (ONS) for this purpose. Domain name service is a resolver that maintains the mapping from URL to IP address or vice versa, while object name service translates EPCs to address of Internet repository in EPCglobal architecture framework. Our resolving operations are identical with domain name service, excepting they are configured to translate service identifiers to the corresponding location of service metadata repository to meet our requirements. Resolving process is hidden to users, and it can be considered special software functions, which implement distributed hierarchical translations and return relevant address *f*(*N*) of service metadata repository for given service identifiers *N*. Figure 1 and Figure 3 show conceptual service metadata architecture.

```
<profile name=prototype>
    <sensor number=1>
      <type>flame</type>
      <application>fire alarm</application>
      <peakCount>30mA</peakCount>
      <sensitivities>5000cpm type</ sensitivities >
    </sensor>
    <sensorNode>
      <chipset>atmega 128L</chipset>
      <networkStack> ZIGBEE </networkStack>
      <watchdog> 8sec</watchdog>
   </sensorNode >
   <sensorData >
      <serviceRepresentation>
         <description>Response of Get Temperature</description>
         <responseId>2</responseId>
         <field order="1">
            <description>nodeId</description>
            <type>integer</type>
         </field>
         <field order="2">
            <description>command</description>
            <value>response_get_temp</value>
            <type>text</type>
         </field>
         <field order="3">
            <description>result</description>
            <type>integer</type>
         </field>
      </serviceRepresentation >
     <sensorEvent>
          <primitiveEvent>
               <description> This is a flame sensor </description>
          <eventLevel> 4 </eventLevel>
          <value>Flame</value>
          </primitiveEvent>
   </sensorData>
   </profile>
```

Table 1 Sensor metadata example

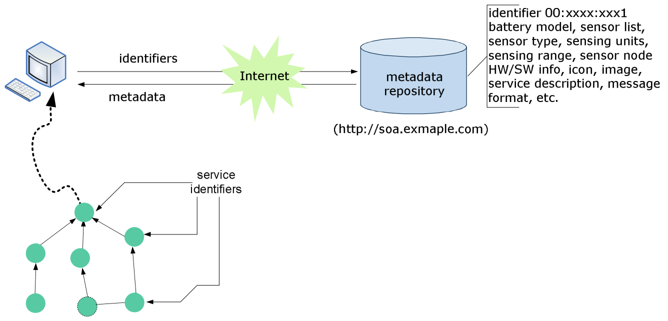


Figure 1 service metadata repository

C. *Service planning*

As results of service discovery and browsing, available service list from sensor nodes are automatically detected, and their capabilities (sensor types, sensing accuracy, sensing units, duty cycle, etc.) are graphically displayed with icons. The icon is also provided by service metadata repository, and the user can set the logical name of each sensor node to ease configurations.

Events produced by one sensor node are internally linked to other events. For example, services using flame detection, temperature/humidity sensor, smoke sensor, and IR sensors are used to composite complex event that attempt to deal with "fire alarm" or "intruder event". User may connect multiple fire detection services to SMS alarm service in order to notify emergency when fire detection service returns fire events. An intuitive GUI allows user to easily configure composite events, while encapsulating complex rule processing internally. The actual realization of composite events is hidden to users. The complexities of complex event and variety of supported rules are dependent upon applications, and detail complex event processing is not addressed here due to page limits. Figure 2 shows service planning.

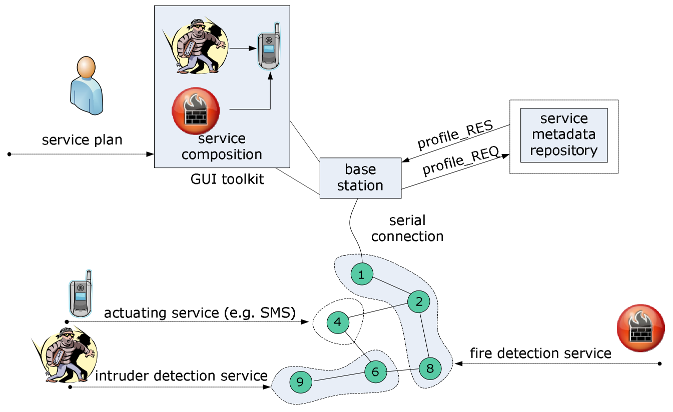
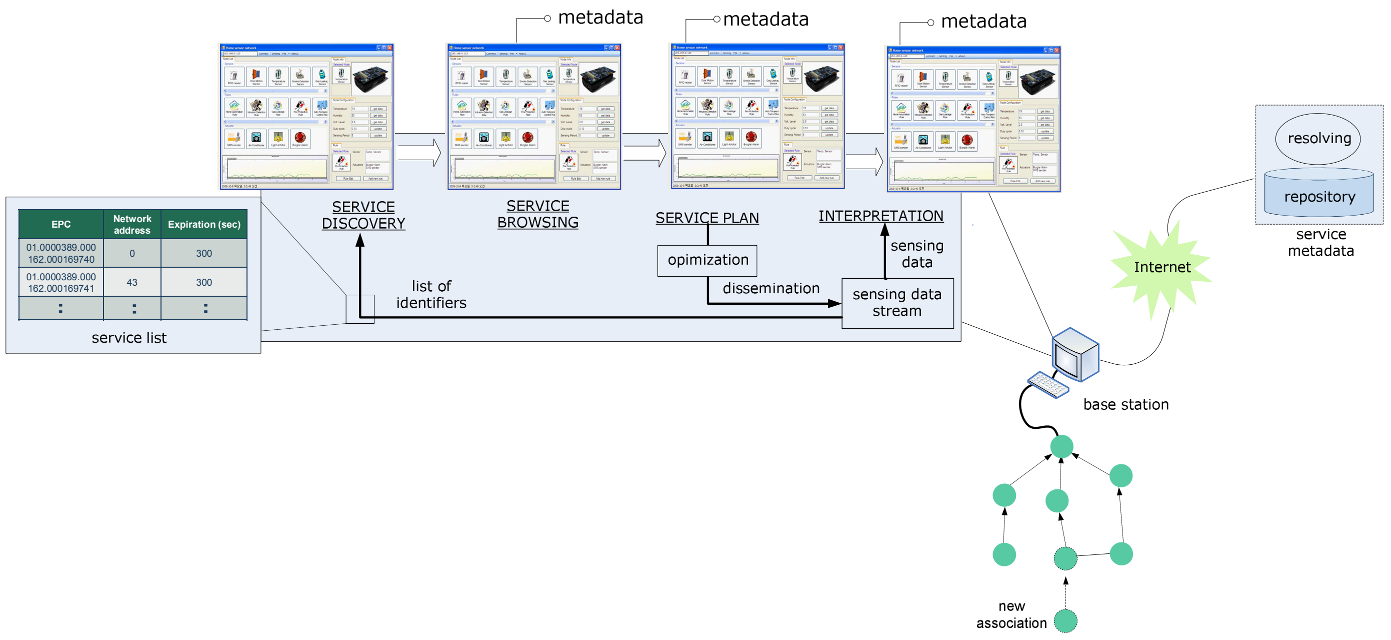


Figure 2 service planning



EPC	Network address	Expiration (sec)
01.0000389.000 162.000169740	0	300
01.0000389.000 162.000169741	43	300
:	:	:

Figure 3 service oriented sensor networks

D. Service optimization

The service plans which are defined by users are internally changed into message primitives for query disseminations. Because available list of services and their capabilities are known by service discovery and metadata browsing, various optimizations can be applied according to sensor types, cost of query and sensing, arithmetic processing, and network topology. For example, user can send query primitives to only relevant sensor nodes (e.g., sensor nodes which have target services such as IR sensors).

We used a communication method selection algorithm that estimates the routing cost of broadcasting and unicasting with considerations of number of receiving sensor nodes. When a base station sends the same packet to more than threshold *a* number of nodes in a network, broadcasting a packet is more efficient than unicasting multiple packets. On the other hand, unicast is more efficient than broadcasting when the sink node sends the same packet to less number of nodes than threshold *a* in the network. The proposed selection algorithm provides an efficient cost evaluation for communications.

The *Select_Method()* function described in Table 2 is the algorithm for the sink node to select the more efficient way for the given destination set. It compares the route cost between unicast and broadcast using the number of required hops to transmit as the metric. For simplicity, this function assumes that the network is stable enough to communicate without failure. Therefore, the required number of hops to propagate a broadcast packet is the same as the total number of nodes in the network.

Since calculation of route cost depends on the network protocol, the *Calculate_Hops()* function is proposed based on the ZigBee network protocol. In ZigBee protocol, every potential parent is provided with a finite sub-block of the address space, which is used to assign the network addresses to its children.

Table 2. Algorithm to select communication method

Select_Method(dstAddr)

```
Input: Set of dstAddr ($d_1$, $d_2$, …, $d_k$), total number of nodes n
Output: Unicast or Broadcast
Begin
 1.totUnicastHops = 0
 2.totBroadcastHops = n
 3.for i = 1 to k
 4.    totUnicastHops += Calculate_Hops($d_i$, 0, 0)
 5.end for
 6.if (totUnicastHops < totBroadcastHops)
 7.    return Unicast
 8.else
 9.    return Broadcast
10.end if
End
```

Given nwkmaxChildren (*Cm*), nwkmaxDepth (*Lm*), and nwkmaxRouters (*Rm*), we can compute the function *Cskip(d)* as the size of the address sub-block distributed by each parent at depth d as follows:

$$Cskip(d) = \frac{1 + Cm - Rm - Cm \cdot Rm^{Lm-d-1}}{1 - Rm}$$

For example, the kth router and nth end device shall be assigned the network address by their parent at depth *d* as in the following equation.

$$A_k = A_{parent} + Cskip(d) \cdot (k-1) + 1 \quad (1 \le k \le Rm)$$

$$A_n = A_{parent} + Cskip(d) \cdot Rm + n \quad (1 \le n \le Cm - Rm)$$

Because of the hierarchical address assignment scheme, any device with address A at the depth d has the destination device with address D if the following equation is satisfied.

$$A < D < A + Cskip(d-1)$$

This *Calculate_Hops()* function in table 2 is the algorithm to get the depth of the destination using the characteristic of ZigBee tree routing [11]. This function is a recursive function that has the arguments *dstAddr*, *startAddr*, *curDepth*. A *startAddr* is the address of the ancestor node at *curDepth* for the given destination *dstAddr*. It is started with *startAddr 0* and *curDepth 0* by calling from the *Select_Method()* function, and returns the route cost, that is, a required hops to transmit to the given destination *dstAddr*.

Table 3. Algorithm to find a hop counts to the destination

Calculate_Hops(dstAddr, startAddr, curDepth)

```
Input: dstAddr, startAddr, curDepth
Output: depth_dstAddr
begin
 1.if (dstAddr = startAddr)
 2.   return curDepth
 3.else
 4.   for i = 1 to Rm
 5.      if (dstAddr is within address space of i^th router)
 6.         return Calculate_Hops (dstAddr, i^th router, curDepth+1)
 7.      end if
 8.   end for
 9.   if (Cm-Rm > 0)
10.      if (dstAddr is the end device of startAddr)
12.         return curDepth+1
13.      end if
14.  end if
15.end if
end
```

* *Address space of i^th router* = (startAddr+Cskip[curDepth]*i+1 , startAddr+Cskip[curDepth]*(i+1)+1)
* *Address space of end device of startAddr* = (startAddr+Cskip [curDepth]*Rm+1, startAddr+Cskip[curDepth]*Rm+Cm-Rm)

E. Service Interpretations

Sensor nodes which get request messages from a base station respond to it. The response messages depend on sensor data representations defined in service metadata, but it basically includes service identifiers, subscription numbers to distinguish them among multiple responses. It is possible for users to make historical queries because the base station parses response messages and stores service results such as sensing data or events. If sensing value in response messages needs calibration or additional information processing service metadata gives instructions for it. As a simple example, raw sensing value read from analog to digital converter can be calibrated into temperature value with user readable Fahrenheit degree.

IV. Conclusion

We presented a service oriented wireless sensor networks toolbox, which encapsulated complexities of sensor networks and enabled simple service compositions using an intuitive GUI. We utilized sensor networks as a collection of services that were discovered and coordinated. Service metadata that self-described both service capabilities and interfaces were defined and exposed via our service metadata repositories. A service oriented sensor network toolbox consisting of sensor networks, an intuitive GUI application that helped users to composite various roles were presented.

We stored metadata in Internet servers and provided a clue to find service metadata from service identifiers. Based on metadata architecture, we presented a sequence of service composition processes including service discovery, service browsing, service planning and optimization.

References

[1] MP Papazoglou, D Georgakopoulos, "Service-Oriented Computing", Communications of the ACM, 2003
[2] Manish Kushwaha, Isaac Amundson, Xenofon Koutsoukos, Sandeep Neema, Janos Sztipanovits, "OASiS: A Programming Framework for Service-Oriented Sensor Networks", 2nd International Conference on Communication Systems Software and Middleware, 2007. COMSWARE 2007, 7-12 Jan. 2007
[3] Jie Liu1 ,Feng Zhao1, "Service-Oriented Computing in Sensor Networks", Lecture Notes in Computer Science, 2005, Volume 3560/2005
[4] Jeffrey King Raja Bose Hen-I Yang Steven Pickles Abdelsalam Helal, "Atlas: A Service-Oriented Sensor Platform: Hardware and Middleware to Enable Programmable Pervasive", 31st IEEE Conference on Local Computer Networks, 2006
[5] Jaco M. Prinsloo, Christian L. Schulz, Derrick G. Kourie,W. H. Morkel Theunissen, Tinus Strauss, Roelf Van Den Heever,Sybrand Grobbelaar, "A service oriented architecture for wireless sensor and actor network applications", Proceedings of the 2006 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries
[6] Marco Sgroi, Adam Wolisz, Alberto Sangiovanni-Vincentelli, Jan M. Rabaey, "A Service-Based Universal Application Interface for Ad Hoc Wireless Sensor and Actuator Networks", whitepaper, UC Berkeley, 2004.
[7] Ayman Sleman , Reinhard Moeller, "Integration of Wireless Sensor Network Services into other Home and Industrial networks", Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on
[8] Flavia Coimbra Delicato , Paulo F. Pires , Luci Pirmez, Luiz Fernando Rust da Costa Carmo, "A Service Approach for Architecting Application Independent Wireless Sensor Networks", Lecture Notes in Computer Science, 2005.
[9] Flavia Coimbra Delicato, Paulo F. Pires, Luci Pirmez, Luiz Fernando Rust da Costa Carmo, "A Flexible Web Service Based Architecture for Wireless Sensor Networks," icdcsw,pp.730, 23rd International Conference on Distributed Computing Systems Workshops (ICDCSW'03), 2003
[10] Sanjay Sarma, David L. Brock, Kevin Ashton, "The Networked Physical World, Proposals for Engineering the Next Generation of Computing, Commerce & Automatic-Identification," Auto-ID Center white paper, http://autoidlabs.org, 2006.
[11] T. Kim, D. Kim, N. Park, S. Yoo, T. S. Lopez, "Shortcut Tree Routing in ZigBee Networks," International Symposium on Wireless Pervasive Computing, 2007.
[12] EPCglobal web sites, http://www.epcglobalinc.org/home.
[13] Sameer Tilak, Kenneth Chiu, Nael B. Abu-Ghazaleh, Tony Fountain, Dynamic Resource Discovery for Wireless Sensor Networks" IFIP International Symposium on Network-Centric Ubiquitous Systems (NCUS 2005).
[14] OSGi Alliance, http://www.osgi.org/Main/HomePage