2014 IEEE International Conference on Internet of Things (iThings 2014), Green Computing and Communications (GreenCom 2014), and Cyber-Physical-Social Computing (CPSCom 2014)

QoS Oriented Sensor Selection in IoT System

Shih-Yuan Yu, Chi-Sheng Shih, Jane Yung-jen Hsu
Dept of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan
r01922040@ntu.edu.tw, cshih@csie.ntu.edu.tw,
yjhsu@csie.ntu.edu.tw

Zhenqiu Huang, Kwei Jay Lin
Dept of Electrical Engineering and Computer Science
University of California, Irvine
Irvine, California, USA
klin@uci.edu

***Abstract*—Systems built for future Internet of Things (IoT) may have a large number of intelligent objects with sensing, actuating, and computing capabilities. In a specific context, a smart application deployed in the sytem needs to meet QoS requirement in order to provide a good experience for users. The Wukong middleware use service oriented paradigm to help application developers define the logical functionalities using Flow Based Programs (FBP). Wukong will then map the abstraction of an application onto physical smart devices and actuators. In this paper, we study the QoS oriented mapping algorithm for applications with a set of QoS attributes. After developers specify how each attribute contributes to the overall QoS, Wukong will find the best mapping solution according to the requirement. We formalize the problem as a maximum weighted bipartite problem, and present the algorithm based on the Integer Linear Programming model. We show the algorithm performance by simulation.**

***Keywords*-Internet of Things, Middleware, Service-oriented computing, Service Selection, Service Composition, Matchmaking;**

I. Introduction

Internet of Things (IoT) envisions a future that a large number of real-world objects will be integrated through Internet. Due to the advances in sensor technology and the decreased sensor production cost, the growth of sensor deployments has increased over past five years. It was estimated that the number of sensors and actuators will grow to millions and even billions. As a result, interoperability, scalability and flexibility challenges must be addressed carefully [1]. Most smart objects with sensing and actuating capabilities are developed on different platforms and are connected to different networks. Wide heterogeneity of smart objects increases the difficulty of interaction between IoT systems.Application developers need to re-engineer and redeploy their applications, which is considered laborious and time-consuming task, whenever there is a change in environment.

To isolate the platform independent logic application functionality with the physical devices that provide sensing and actuating features, the WuKong project [2], [3] supports a flow-based programming (FBP) model, shown on the right side of Fig. 1, to define the data and control flow among virtual sensing devices. Given an FBP, users can specify policies to impact the system deployment decisions. The WuKong middleware will then select the physical devices to provide specific functionalities under constraints from policies.

Our previous study [4] presents a flexible mapping algorithm that achieves a low overall system energy consumption and long system life-time under location and energy policies on FBP components and devices respectively. However, the model only considers energy consumption as the optimization goal. In real world, the mapping decision for an application should also achieve other runtime QoS objectives like accuracy, response time.

In Fig. 1, we show an IoT system deployed in an indoor environment on the left and an intelligent application on the right. The FBP specifies that a PIR component will trigger the AC control and to turn on the light. If there are two physical PIR sensors capable of providing the functionality, the system may select the one by the front door rather than the one in the kitchen according to the response time QoS. On the other hand, if these two PIR sensors are heterogeneous, the one in kitchen is of high quality and the one by the front door is not working well, the system should use the one in the kitchen for the FBP if accuracy is more important.

In this paper, we introduce a quality score system to hide the complexity and diversity of application QoS, and present a new mapping methodology that takes multiple QoS attributes into consideration and finds the optimal deployment decision with the highest overall application level quality score. We model the problem as a maximum weighted bipartite problem, and solve it using ILP.

The rest of the paper is structured as follows. We review the literature on IoT middleware solutions and related methodology about service selection and composition in Section II. In Section III, we present the background on our WuKong project. The problem description and motivations of our researchis given in Section IV. Section V presents the proposed solution using the Maximum Weighted Bipartite Matching problem model. Performance study using the proposed algorithm is reported in Section VI. Finally we

DOI 10.1109/iThings.2014.37

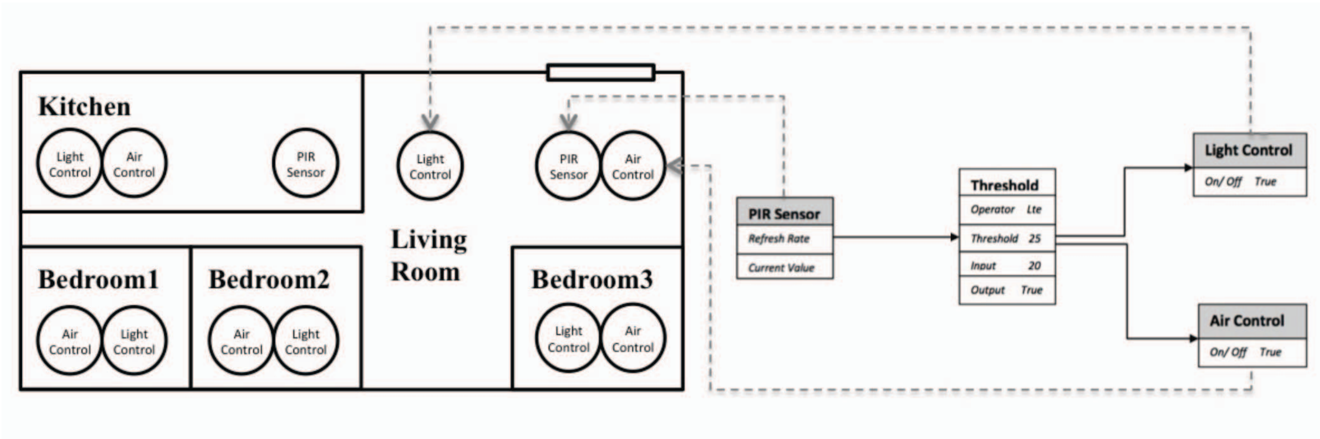

Figure 1. Deployment example between abstract IoT application and target environment

present a conclusion and prospects for future research in Section VII.

II. Related Work

Since the vision of the explosive growth IoT industry, many companies design their own hardware, RF communication protocol, and smart things competing for the market. It brings the problem of interoperability between devices and also the programmability of connected things. Many Internet of Things projects present middleware solutions to improve the programmability of IoT systems. As a toolkit for IoT, middleware can help hide the complexity and heterogeneity of the underlying hardware and network platforms, ease the management of system resources, and increase the predictability of application execution [5], [6]–[8]. MagnetOS and LooCI [9] are two of these middleware designed for wireless sensor network. By adopting philosophy in IoT, WuKong applications are viewed as a virtual flow between smart objects [2], [3]. In WuKong, IoT programmers can interact with IoT systems by translating their requirements into high-level flow of information. Therefore, sensors and actuators can communicate with each other even if they are deployed in different platforms and different networks.

In service computing community, automatic QoS aware service composition [10] and service selection [11] are two active research topics. In the research of service selection, the functionality of distributed system is described by a business process which is composed by a set of abstract services in a directed acyclic graph. The problem is to find a best combination of real services to achieve overall QoS optimization by finding proper services instances from a large set of service candidates with different QoS. The automatic service composition problem is even harder for composition algorithm need to compose a flow of service to meet the both functionality requirement and QoS constraints. In this work, the QoS service is well defined, but the characteristic of every heterogeneous physical sensors is different. Our work define a extendable scoring system to convert them into QoS performance values in different application context, environment and user experience.

In the research of sensor network, lots of researchers devise MAC protocols and routing protocols to improve the QoS of homogeneous sensor network in the terms of energy consumption, reliability and real-time data transmission. Since IoT system is designed to improve qualify of people's life, thus mostly in in-door environment. In such environment, the QoS of a system is no longer determined by network itself, but also impacted by the heterogeneous feature of things and also interoperation between things. Our research mainly take these into attributes consideration, and aim to improve the QoS of intelligent applications itself. All these work inspire us in the aspect of finding the optimal deployment solution for a FBP according to the QoS requirement of an intelligent application in the highly evolvable and reconfigurable WuKong middleware.

III. Background Knowledge

The WuKong IoT system consists of a mixture of distributed embedded node devices and gateway node devices. Each embedded node device is capable of running multiple tasks such as sensing, computing, actuating at the same time, depending on what sensors or actuators are installed. The gateway devices maintain the communication of neighboring node devices via RF communication. WuKong adopts master management module, which plays as a policymaker, to coordinate and manage its underlying sensor node devices. WuKong master program can run on some gateway devices and configure all its manageable devices, making them autonomously cooperate with each other and monitoring them if needed.

In WuKong, flow based programming model is adopted for composing IoT applications. In Fig. 2, users interact with FBP editor tool, and utilize a set of virtual service components, which is called *WuClass* in WuKong, to design an application by organizing the flow of information between them. The mapping compiler in WuKong accepts and translates the high-level language application into a number of machine-dependent codes, and then deploys them onto corresponding devices. Therefore, the node devices can cooperate with each other like the way designed in FBP editor as shown in Fig. 1. In this way, WuKong can support agile development and just-in-time configuration for IoT applications.

A. IoT services

Smart thing with sensing, actuating and computing capabilities can be abstracted as IoT service components, and further utilized for designing IoT applications. In WuKong, we use WuClasses to model IoT services. For WuKong design environment (including WuKong master, FBP editor, node devices), we uses a universal WuClass library for references in order to model an IoT service. Each WuClass has a definition in the library XML file, which make the WuClass manageable for WuKong master. The definition of a WuClass includes,

- The set of functional properties as exposed resources for the system.
- The set of non-functional properties that specify the characteristics of the WuClass. These properties may be updated by WuKong master during run-time monitoring.
- The implementation of *update()* function used to specify WuClass' behavior.

In addition, the definition also include WuClass hierarchy so as to model the relationship between WuClasses. Outdoor temperature sensor service and indoor temperature sensor service can both be classified as temperature sensor service. The functional properties are the resources that a WuClass provides for system, reading value is an example functional properties for temperature sensor service. Two functional properties can be connected if the type of the properties are compatible. Non-functional properties can describe the characteristics of a WuClass service. In IoT system, non-functional properties on each service are needed to be updated as the real-time condition or environment changes for more accurate service specification.

After defining WuClasses, WuObject is the instance to perform functionality of the WuClass. Many WuObjects, or services, are distributed in an IoT system as the left side of Fig. 1 shows. Each node device in WuKong system can have multiple WuObjects running on. When there is any change on the properties in a WuObject, WuKong software triggers the *update()* function of the WuObject and updates other functional properties. For example, when there is change for the input value of threshold component, the update function of threshold component will be triggered and new output value is stored after the computation. Then, the software on each node device will propagate functional property values between WuObjects if the property has been changed.

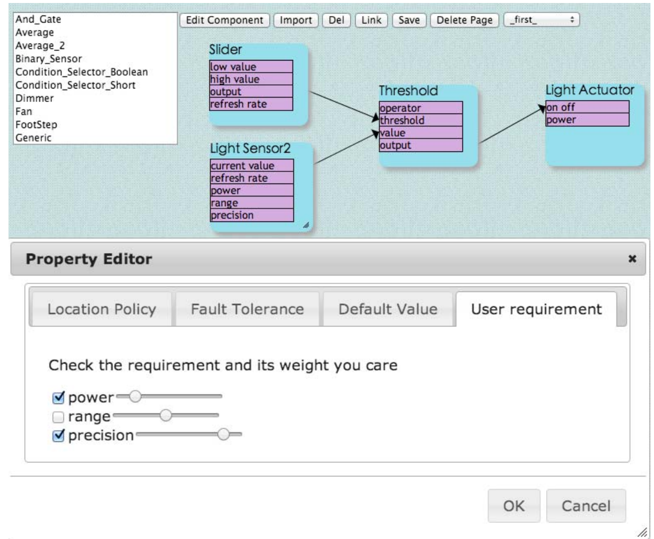

Figure 2. Example of Flow Based Program (FBP) editor

B. IoT applications

Most IoT applications are event-driven by nature, thus we can use the flow of information between virtual service components, which is called service graph, to describe the logics in an application. A service graph is defined as a network of virtual service components, each of which belongs to a service class. The link between service components stands for the flow of information. In WuKong, FBP model implements the concept of service graph using *WuClass* component blocks as Fig. 2 shows.

A WuKong user can define the initial functional property value for each *WuClass*, such as activation threshold value and refresh rate. After a FBP is deployed, WuKong is designed to monitor and identify what property values are best for system performance and application preference and then automatically refine the settings and update functional properties. A WuKong user can also specify the user preference for each *WuClass* as Fig.2 shows. For each component, the sliders in property editor can be adjusted and used to specify how each non-function property contributes to overall quality of service performance. For example, Property ”energy efficiency” is weighted heavier than ”precision”, when the developer prefers an energy-efficient service even when the precision is not good enough.

IV. Problem Description

WuKong is responsible to deploy all WuClasses on FBP to their corresponding WuObjects, each of which is host by physical device in IoT systems. Among all correct mappings, how to select a good one is the problem we are interested. In Fig. 1, the application find its best mapping is to map the

PIR WuClass to the PIR sensor near the door instead of the one in the kitchen.

A. IoT Service Matchmaking

Each IoT application implemented in FBP model can be defined by a directed acyclic graph (DAG) $F = (C, L)$ with user preference P where,

- C is the set of WuClasses.
- L is the set of link that stands for the flow of information. WuClasses can be connected via the functional properties on each WuClass.
- P is user preference set for WuClasses. For each WuClass C_i, user can decide the values in weight set $P(C_i)$. For each component C_i, we let $\sum_k P_k(C_i) = 1$. Moreover, each element $P_k(C_i)$ represents the proportion of emphasis that an user puts for a specific QoS attribute.

An IoT systems M consists of a set of available services S, and a set of property set Q to describe the characteristics of each service. The services are the instances of virtual service components, which we called *WuObject* in WuKong. Through automatic service discovery and capability identification, WuKong can discover all the WuObjects which are provided by some physical devices in M. Because of different real-time and hardware condition for each WuObject, a vector $Q(S_j)$ is used to describe it. The value of each element in $Q(S_j)$ represents a specific QoS attribute and is normalized to be ranged from 0 to 1 when in our problem.

In WuKong, a WuObject S_j can be matched to a WuClass C_i if S_j is capable of providing the needed functionalities for C_i. The class hierarchy defined in WuKong master help determine if a WuClass can be mapped to a WuObject. For example, a WuClass "Temperature" can not only map to "Temperature" WuObject, but also be mapped to "Wall Temperature Sensor" or others. Based on the hierarchical definition of WuClasses, WuClass C_i and WuObject S_j can be matched if and only if they are related in WuClass hierarchy definition. Thus, we defined the matching score $y(C_i, S_j)$ between C_i and S_j as Eq. 1 with class hierarchical relation, where we set $\alpha = 0.6$ and $\beta = 0.4$ here.

$$y(C_i, S_j) = \begin{cases} 1 - d(C_i, S_j), \text{if } C_i \text{ is same class with } S_j \\ \alpha(1 - d(C_i, S_j)), \text{if } C_i \text{ is super class of } S_j \\ \beta(1 - d(C_i, S_j)), \text{if } C_i \text{ is sub class of } S_j \\ 0, \text{otherwise} \end{cases} \tag{1}$$

We evaluate the QoS performance of S_j when map it with service component C_i under user preference $P(C_i)$ using the weighted distance function defined in Eq. 2. Each element in user preference of WuClass C_i, $P_k(C_i)$, represents how a QoS parameter $Q_k(S_j)$ contribute to overall QoS performance $d(C_i, S_j)$.

$$d(C_i, S_j) = \sqrt{\sum_k P_k(C_i)(1 - Q_k(S_j))^2}, \tag{2}$$

B. Problem Definition

Given an IoT application implemented in FBP $F = (C, L)$ with P and an IoT system M consists of S and Q, we want to find the best user-fit QoS-aware matchmaking to deploy the FBP. We define the deployment matrix x to specify the relationship between WuClasses C and WuObjects S, where variable $x_{ij} = 1$ denotes that C_i has been matched with S_j, $x_{ij} = 0$ to denote C_i has not being matched to S_j. Therefore, the total matching score for each service component C_i can be computed by Eq. 3.

$$\text{Score}(C_i) = \sum_j x_{ij} \times y(C_i, S_j) \tag{3}$$

The objective function can be defined as Eq. 4. We want to find the matching matrix x so that the overall total matchmaking score over FBP F is maximum.

$$\underset{x}{\operatorname{argmax}} \sum_i \text{Score}(C_i) \tag{4}$$

V. User-Oriented Matchmaking Scheme

This section presents the user-oriented QoS-aware matchmaking scheme. Firstly, we show how we model the problem to be maximum weighted bipartite matching problem. Then we introduce the integer linear programming (ILP) formulation for the problem.

A. Maximum Weighted Bipartite Matching Problem

We introduce the Maximum Weighted Bipartite Matching problem. Given a edge-weighted bipartite graph $G = (V, E)$ with partition (A, B) and a weight function $W : E \to R$, we want to find a matching of maximum weight where the weight of matching M is given by $\sum_{e \in M} W(e)$. The matching M in G is a set of pairwise non-adjacent edges, which means no two edges share a common vertex.

We model the service matchmaking problem to maximum weighted bipartite matching problem, which is known as the assignment problem. We consider an edge-weighted bipartite graph $G(V, E)$ and a weighting function W, with partition (A, B) as Fig. 3 shows, where

- Partition A is the set of WuClass in FBP F,
- Partition B is the set of WuObject in IoT system M.
- Edge set E is the pairs between partition A and B.
- Weighting function W is defined as the weighted distance calculated by Eq. 2.

Given FBP $F = (C, L)$ with user preference P and IoT system M which consists S and Q, we firstly construct the corresponding edge-weighted bipartite graph $G(V, E)$ and W, then we can further apply algorithms to find Maximum

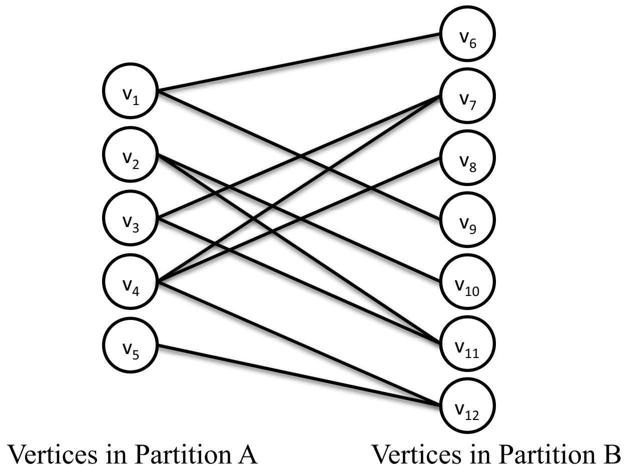

Figure 3. Edge-weighted Bipartite graph $G(V, E)$ and weighting function W with partition (A, B) for solving service matchmaking

Weighted Bipartite Matching on graph G. Algorithm 1 shows how we construct the graph G, which is a complete edge-weighted bipartite graph. We assign the matching score $y(C_i, S_j)$ to be edge's weight even when the matching score is zero.

Algorithm 1 Edge-Weighted Bipartite Graph Generation

Input: FBP $F = (C, L, P)$ and IoT system $M = (S, Q)$
Output: An edge-weighted bipartite graph $G = (V, E)$ and weighting function W

```
for all service component C_i in C do
    create new vertex v and add to partition A and V
    for all service S_j in S do
        create new vertex v' and add to partition B and V
        add edge e = (v, v') to E, and assign the weight
        W(e) = y(C_i, S_j)
    end for
end for
```

B. Problem Analysis

Finding the best user-oriented QoS-aware service matching matrix x between FBP F and IoT system M can be equal to finding the maximum weighted bipartite matching M on the edge-weighted bipartite graph G. In this paper, as the definition of matching, we address one-to-one matching between partition A and B. However, many-to-one and one-to-many mapping could also exist in service selection, we leave them as our future discussion.

Every WuClasses has its best fit WuObject by considering user preference. To compare all possible combinations to find the best solution, we have to compute m^n combinations, where n is the number of WuClass and m is the number of WuObject. However, we can find some heuristic method to efficiently solve service matchmaking. If we traverse all the *WuClasses* in partition A, and choose the most heavier edge for its matching. We can acquire a solution in $O(n \times m)$ time complexity, but it's not optimal solution because in our problem setting, multiple *WuClasses* can have the same current best *WuObject* to match. Therefore, we are interested in solving service matching problem by using bipartite matching algorithm to find the best user-fit solution efficiently since the number of available *WuObjects* can be huge in IoT systems.

C. ILP Formulation

After the edge-weighted bipartite graph $G = (V, E)$ and weighting function W are generated, we apply integer linear programming (ILP) to formulate maximum weighted bipartite matching. Given bipartite graph G, we want to find the matching matrix x where,

$$max \sum_{i,j} x_{ij} \times W(A_i, B_j) \tag{5}$$

subject to:

$$\sum_{i} x_{ij} = 1, \forall 1 \leq i \leq |A| \tag{6}$$

$$\sum_{j} x_{ij} \leq 1, \forall 1 \leq j \leq |B| \tag{7}$$

Eq. 6 ensures that each *WuClass* can find its matching *WuObject* in S because it can be a failed matchmaking if not all the service components on a FBP can be mapped to system. Eq. 7 ensures that each *WuObject* can only be matched by one *WuClass* in an IoT system since we already assume that services, *WuObjects* are not sharable in our problem setting.

VI. Performance Study

In this section, we show experiment results to evaluate proposed ILP for solving service matchmaking. We compare ILP with baseline method, which adopts greedy matching algorithm to greedily find best matching for each component. We show how we set up simulation testbed, the consideration for determining system parameters and performance evaluation index, and performance result.

For an IoT system with a set of available services, we generate simulated systems for our testbeds as follows. For the WuClass hierarchy, we randomly generate a tree-like structure to model the relationship between WuClasses. Then, we randomly generate WuObjects of different class. For each WuObject S_i, we generate a QoS attribute set $Q(S_i)$ with 10 QoS parameters for describing the service, where every element in the set range from 0 to 1. We then use JGraphT library to generate a random flow based program as target IoT applications, including 1000 instances of random topology. For each WuClass component C_i on

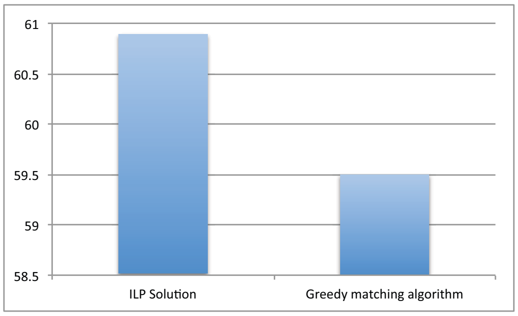

Figure 4. Performance Comparison between ILP and Greedy Matching Algorithm (With 100 components on FBP)

FBP, we randomly generate a weight set $P(C_i)$ with the same size to QoS attribute set, 10, as user preferences. After the weight set is generated for each FBP component, we normalize each element in the weight set so that the total sum of all element in the weight set will be 1.

In this paper, the data for 100 WuClasses in FBP and a system with 1000 WuObjects is reported. The performance evaluation is conducted on an Apple MacBook Air, with a 1.8GHz Intel Core i5 processor and 4 GB RAM. We compare the performance with ILP and Greedy Matching, and we evaluate the performance by the total matching score we defined in Eq. 4.

As Fig. 4 shows, after 1000 simulation, we find out that ILP can find a better user-fit QoS matching for the application, outperforms the greedy matching algorithm. However, ILP takes more time to compute the result, which may not be scalable for the setting of IoT system. This inspires us to investigate the improved greedy-based matching algorithms that can efficiently solve service matchmaking in IoT systems.

VII. Conclusion and Future Work

As the number of IoT devices grows, it becomes challenging to deploy applications through the network. To make programming on IoT simple is important since we have to engage everyone in interacting with IoT network in the future by a user-friendly tool. Automatically finding a good mapping between abstract application and IoT systems is important for intelligently helping programmers deploy their applications. By using WuKong, we raise the level of programming abstraction so that users can only focus on their logical design of an application. We formulate the service matchmaking problem to maximum weighted bipartite matching problem, and we compare the performance between greedy matching and ILP solution. We find out ILP solution is optimal for the service matchmaking but consume more time which may not be scalable in large-scale IoT systems. Improving the greedy based service matchmaking is a future direction of this research. In addition, we should investigate automatic composition module to address a more complicated application, with the goal of supporting more users to interact with IoT systems with ease.

Acknowledgment

This work was also supported by National Science Council, National Taiwan University and Intel Corporation under Grants NSC102-2911-I-002-001 and NTU103R7501.

References

[1] T. Teixeira, S. Hachem, and N. Georgantas, "Service Oriented Middleware for the Internet of Things : (Invited Paper)," vol. 257178, no. 257178, pp. 220–229, 2013.

[2] K.-J. Lin, N. Reijers, Y.-C. Wang, C.-S. Shih, and J. Y. Hsu, "Building Smart M2M Applications Using the WuKong Profile Framework," *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, pp. 1175–1180, Aug. 2013.

[3] N. Reijers, K.-j. Lin, Y.-c. Wang, C.-s. Shih, and J. Y. Hsu, "DESIGN OF AN INTELLIGENT MIDDLEWARE FOR FLEXIBLE SENSOR CONFIGURATION IN M2M SYSTEMS."

[4] Z. Huang, K.-J. Lin, and A. Han, "An Energy Sentient Methodology for Sensor Mapping and Selection in IoT Systems," *2014 IEEE International Symposium on Industrial Eletronics*, 2014.

[5] M.-m. Wang, "Middleware for Wireless Sensor Networks : A Survey," vol. 23, no. 2006, pp. 305–326, 2008.

[6] R. Barr, J. C. Bicket, D. S. Dantas, B. Du, T. W. D. Kim, B. Zhou, and E. G. Sirer, "On the need for system-level support for ad hoc and sensor networks," *ACM SIGOPS Operating Systems Review*, vol. 36, no. 2, pp. 1–5, Apr. 2002.

[7] C.-l. Fok, G.-c. Roman, C. Lu, and S. Louis, "Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications."

[8] D. Hughes, K. Thoelen, W. Horré, N. Matthys, J. D. Cid, S. Michiels, C. Huygens, and W. Joosen, "Looci: A loosely-coupled component infrastructure for networked embedded systems," in *Proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia*, ser. MoMM '09, 2009, pp. 195–203.

[9] J. Bao, Y. Ding, and H. Hu, "A New Service Selection Algorithm in USPIOT Nanjing University of Posts and Telecommunications," no. 1, pp. 6–10, 1999.

[10] Z. Huang, W. Jiang, S. Hu, and Z. Liu, "Effective Pruning Algorithm for QoS-Aware Service Composition," in *2009 IEEE Conference on Commerce and Enterprise Computing*. IEEE, Jul. 2009, pp. 519–522.

[11] Y. Yan, B. Xu, Z. Gu, and S. Luo, "A QoS-Driven Approach for Semantic Service Composition," *2009 IEEE Conference on Commerce and Enterprise Computing*, pp. 523–526, Jul. 2009.