Service Discovery and Composition in Body Area Networks

Matteo Coloberti
D.I.Co.- University of Milan
Nokia Siemens Network, Italy
coloberti@dico.unimi.it

Clemens Lombriser
Wearable Computing Lab
ETH Zürich, Switzerland
lombriser@ife.ee.ethz.ch

Daniel Roggen
Wearable Computing Lab
ETH Zürich, Switzerland
droggen@ife.ee.ethz.ch

Gerhard Tröster
Wearable Computing Lab
ETH Zürich, Switzerland
troester@ife.ee.ethz.ch

Renata Guarneri
Nokia Siemens Network
Milan, Italy
maria.guarneri@nsn.com

Daniele Riboni
D.I.Co.- University of Milan
Milan, Italy
riboni@dico.unimi.it

ABSTRACT

In pervasive environments, Body Area Networks (BANs) are characterized by the mobility of their users. BANs can continuously interact with each other, thus enabling the provision of new applications and services at runtime. New complex services can be provided by composing simpler services available on neighbouring network nodes. However, since the topology of BANs is continuously changing due to users' movements, it is unfeasible to specify *a-priori* all possible configurations under which a given complex service can be composed. In order to address this issue, we introduce a two–layered service discovery and composition architecture, that proactively notifies a distributed service directory with changes in service availability. In order to cope with the network mobility and intermittent connectivity, our approach is to cluster nodes in the sensor network based on their connectivity patterns. We use a multi–agent state machine to recognize the availability of complex services and to provide them. Our solution is validated by a prototype implementation of our architecture, by the study of the statistical model of complex services, and by experimental evaluations.

1. INTRODUCTION

Wireless Sensor Networks (WSN) enable new ways of gathering contextual information related to mobile users in order to provide them with just-in-time context-aware services. In a heterogeneous network with devices attached to the body and integrated into people's environment, each sensor can contribute with a specific set of sensors and computation services that process sensor data and turn them into useful information for the user of the distributed system.

Within the e-SENSE project [9] we develop an architecture for capturing ambient intelligence for beyond third generation (B3G) mobile communication networks. We aim to connect mobile devices having B3G connection to sensor networks on the body and in the environment in order to provide context-aware services to users. The large heterogeneity of sensors and services is abstracted by an integration middleware layer, which, among other functionalities, composes complex distributed services by assembling a number of *simple services* offered by the single sensor nodes. Such a *complex service* is essentially an algorithm that processes sensor data through various steps and composes them into higher–level context information. The simple services provide algorithmic building blocks.

In e-SENSE, the distributed execution of complex services is obtained by decomposing them into simple services, or tasks. These tasks are interconnected to form *service task graphs* (TGs) that are executed by our Titan task graph execution framework [8]. This paper describes the service discovery and composition mechanism, which utilizes *Finite State Machines* (FSM) to provide a light architecture to determine what kind of complex services are executable in the connected service clusters.

Service discovery (SD) issues in the context of Mobile Ad Hoc Networks (MANET) have been discussed in the literature; recent surveys [10] classify protocols into strategies for Service Discovery and Service Information Accumulation. Service Information Accumulation uses different methods to store service information: central or distributed service directories, and also directory-less protocols. An example of a solution using service directories is PDP [5], a fully distributed protocol that gives priority to messages of low capacity devices and uses GSDL for service description.

Various macroprogramming models for WSNs in the form of service task graphs have been developed: e.g. Titan [8], OASiS [7], or DFuse [6]. Those architectures show the feasibility and soundness of the approach, but do not describe how applications can be composed in different ways to adapt to the dynamic nature of Body Area Networks. Service composition for WSN follows different approaches based on architecture requirements, aggregation algorithms and language descriptions. Previous work on task graphs has been proposed by [2]. In that work service composition is performed upon application execution request by a global search algorithm across the WSN.

The distinct advantages of our approach for Body Area Networks are that *i)* It can adapt to the very dynamic network topology encountered by mobile users; *ii)* It is a light-weight architecture which requires much less processing power than other solutions based on ontological reasoning; *iii)* Our approach enables the composition of complex services according to the current users' situation and to their QoS requirements specified with different TGs.

2. ARCHITECTURE OVERVIEW

Body Area Networks (BAN) are characterized by a large variety in the number of wireless sensor nodes connected to the network. As users are in movement, the network topology is constantly changing. As a consequence, the type and

availability of complex services may change over time. However, dynamic composition of services as well as storing all complex service configurations and evaluating the optimal one in every situation is challenging in resource-constrained wearable devices. In our work for the e-SENSE project [9] we thus proposed to store a database of complex services in a *Task Graph Database* (TGD) on a server connected to the BAN through a B3G connection and a gateway device, e.g. a mobile phone. The Wireless Sensor Network nodes on and around the body are clustered and a service directory collects the simple service information for every such cluster in a local database. Via a B3G gateway the sensor cluster notifies the TGD about changes in service availability. If new complex services are executable in the cluster, the TGD can register the corresponding service task graph description in the service directory in the cluster, thus making it available for instantiation in the BAN. Figure 1 highlights the

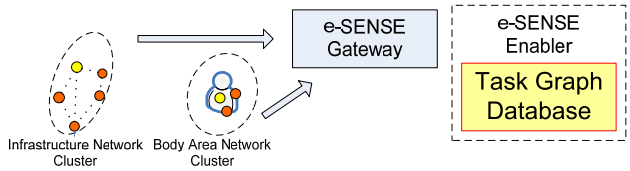

Figure 1: The e-SENSE project architecture overview.

e-SENSE architecture. It contains the e-SENSE TGD, the e-SENSE gateway (e.g. a mobile device), and the e-SENSE WSNs organized into clusters of sensors. In this paper we focus on the interaction between BANs and the *service discovery* protocol, in order to publish newly available services to the *service directory* of the server platform.

2.1 Simple Service Discovery

Every sensor node keeps a *service pool*, where all services available on the node are registered. The number and type of those services depend on the capabilities of the sensor node. In order to achieve scalability, sensor nodes are grouped into service clusters, in which one node is elected as the clusterhead and service directory, and collects the service information of all nodes in its cluster. Body Area Networks are very dynamic networks. In order to provide a stable basis for the execution of complex services, the network nodes must be clustered into groups of nodes that will remain together for an extended amount of time. If other service clusters appear, a node registers its services on the node with which it has been connected for the longest time. This decision ensures that two service clusters crossing each other (e.g. two people walking by each other) remain stable, i.e. nodes do not change their service directory.

When a node is started up, it first tries to connect to an existing clusterhead known by its neighbors. If this is not possible, it tries to build its own cluster, assuming the role of the clusterhead. As long as the new cluster is not stable, it tries to join another cluster, giving up its role. On the contrary, if the new cluster becomes stable, the node tries to conserve its role. Cluster leaves in reach of multiple clusterheads periodically evaluate connection statistics, such as hop-count and connection time to decide in which cluster they will stay.

The service directory is instantiated on the clusterhead and receives the service registrations of the leaf nodes. It maintains a list of neighboring clusterheads and the path to the e-SENSE gateway. Service registrations need initially to be covering the whole service pool of a service node, and after that only periodic update messages are required, which ensure that service registrations do not time out. When new service types become available or disappear within the cluster, the service directory sends an update message containing the modifications to the TGD via the gateway.

2.2 Services as Task Graphs

In our approach, we distinguish between *simple services* and *complex services*. Simple services may include sensors, data processing functions, or complete algorithms. Each simple service is identified by the global unique identifier of its functionality. When the simple services are instantiated, a set of parameters can be supplied to adapt the functionality to application requirements Complex services are composed by a

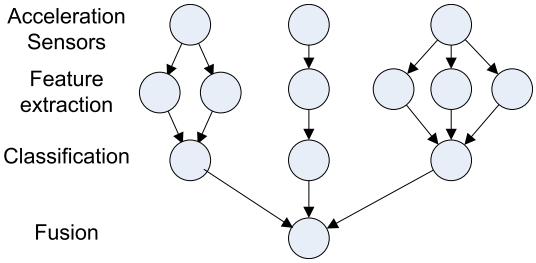

Figure 2: Example of a complex service for activity recognition, showing simple services available on sensor nodes for each step of the algorithm.

number of simple services interconnections, which together form a service task graph. Simple services can thus be seen as building blocks for algorithms described as complex services. An example of a complex service for the recognition of human activity following the approach of [1] is given in figure 2. Acceleration data is sampled from sensors, features are extracted from the signal, which then are used to classify the motion of the sensor node, such as a gesture by a hand, or the motion of a tool. Classification events from multiple sensors are used to determine the overall human activity by fusing events generated in the network.

Using services and service task graphs as a programming model it is possible to abstract from the details of implementations on the actual sensor node hardware. This is especially important when considering heterogeneous sensor networks with e.g. different types of sensors, microcontrollers, or transceivers. The use of the service abstraction relieves from the burden of developing code for each specific sensor node, since the operation of deploying services to specific devices is demanded to the framework. By supporting a range of different implementations of a complex service, the execution can be adapted to the sensor node number and types currently available in the network. In the case of Titan, the replacement of the service task graph requires only a few miliseconds [8].

3. THE TASK GRAPH DATABASE

The task of the TGD is to determine which complex services can be executed on the connected sensor network. From an high level of abstraction, the TGD is composed by a set of Finite States Machines building a multi-agent hierarchical

structure that can aggregate simple services to provide a complex service. Every agent has two main duties: *i)* to recognize newly available complex services and *ii)* to recognize when a complex service is no more available. The TGD will react by publishing or removing the new complex service to the sensor network service directory. In order to provide complex services, the TGD receives updates from the WSN service directories which report the availability of services and resources involved in the computational process of a complex service. All these service updates reach the TGD as registration or deregistration messages of simple services. The main characteristics of the TGD are: *i)* Each agent in the TGD is responsible for a complex service and receives the respective messages arriving from the WSN. *ii)* The TGD reacts to a sequence of registration and deregistration messages; each message should be processed only by the interested agents. *iii)* When an agent recognizes a new complex service, the TGD advertises the newly available service to the WSN service directory. *iv)* When an agent recognizes a service that cannot be provided anymore, the TGD issues a removal instruction.
According to the description of a complex service, we could assume that in the event of a deregistration message, the related simple service could be replaced by another implementation of the service of the same type. Thus the TGD must be aware of service replacements. Another example would be that an implementation of a simple services for a powerful device could be replaced by a complex service assembling multiple lower-profile simple services. The overall goal is to achieve a lightweight and dynamic architecture of the TGD. Agents inside the TGD can be added or removed easily, supporting stable service composition even for highly mobile sensor networks, and tracking simple service availability and possible replacements in case of disappearing nodes.

3.1 TGD Architecture

The TGD architecture is based on a multi-agent structure generally referring to a refinement of a congregation [3] model where each agent follows a producer-consumer paradigm; each agent is an event-driven state machine that can take the role of a consumer, thereby reading registration and deregistration events in order to compose services. The agent can act as a producer as well and issue messages of services which can in turn be consumed by other agents providing complex services composed of the first ones. Within the large variety of multi-agent structures we have chosen the congregation model as it better reflects the dynamic interactions of the TGD, in fact agents forming congregations need to interact only with some other subset of the agents in the agent population. To understand the real benefit of a congregation structure, we consider a scenario where sensors with different capabilities are deployed on the body and in the environment. Only a few nodes in the network (e.g., mobile phones or PDAs) have higher capabilities; hence, they can provide more computationally intensive services and higher reliability. We assume that in the WSN there are multiple instances of each simple service, deployed on different low-profile nodes. As a consequence, complex services involving just low-profile simple services may be discovered with lower probability due to message loss, but will be available in more different configurations, while complex services making use of high-profile simple services are found with high reliability, but suffer from the fact that they depend on the availability of the powerful devices offering those services.
The TGD hierarchical structure of agents intercommunication mainly depends on agent duty; in fact, different agents act as a group based on which events they consume and produce. The TGD keeps track of the internal agents structurereacting on events and dynamically updating only related agents.

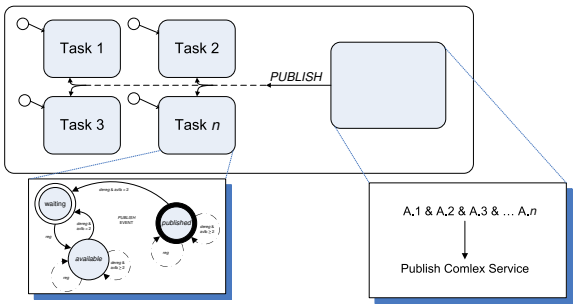

Figure 3: An agent contains for each service it needs an FSM keeping track of service availability.

An agent can be described as a set of concurrent tasks, doing so the overall agent reaches its final state and advertises a new complex service when all its substates have reached their own final state. The architecture depicted in Figure 3 is a FSM with concurrent substates describing a complex service composed by n simple services. Consequently there are n substates, one for each simple service within the main FSM which oversees the complex service availability. In case all n substates become available, the agent generates a *PUBLISH* event that implies the following behaviors: *i)* the new complex service recognized is published to the Service Directory, *ii)* the agent generate a state transition from *available* to *published*, and *iii)* the agent sends a registration message to other agents waiting for this service.

4. EVALUATION

In this section we evaluate our service discovery and composition approach by first derive a statistical model about the number of simple and complex services available in a network. In a second step we simulate the complete approach to analyze its performance.
The service clustering algorithm introduced in section **??** has been simulated with 50 nodes in an area of 100x100 units, each node having a omnidirectional transmission range of 5 units. Message loss has been introduced when nodes are sending too many messages. The nodes have been arranged in clusters with an average of 5 nodes representing Body Area Networks of individual persons. As suggested by [4], the simulation was based on a Reference Point Group Mobility (RPGM) model with random movements of the clusters as well as a random motion within the group. Figure 4 shows the average increase in service formation during the cluster formation group for 100 runs. The delay between the time a node joins a cluster and the time it registers at the service directory is clearly visible. Our model initially overestimates the number of services that should be available. The simulation gradually approaches that limit as more and more services are registered by the sensor nodes at the service directory. The slow increase results from the fact that not all services registered at the node can be sent to the service directory within a single message, and that some registration

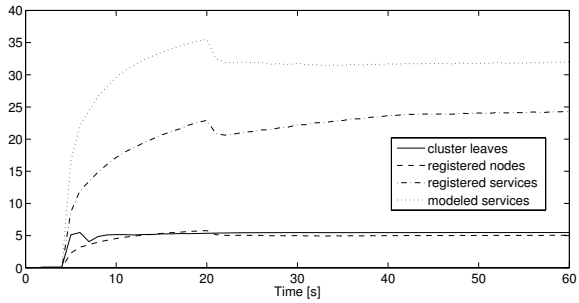

Figure 4: Example of service formation in a cluster. The number of services available on all connected cluster leaves, and the actually registered services at the cluserhead vs. a probabilistic model

messages get lost and are resent only after a certain time.
For the simulation tests we used 40 different complex service descriptions composed of simple services detected by the service discovery protocol of the WSN simulation. For each complex service, we generated an agent compliant to the concurrency model described in section 3.1. Each agent is composed for a different service task graph of a minimum of 3 and a maximum of 20 services. In the beginning, each agent is "waiting" for event notifications. Only after it has reached its final state, it publishes a new complex service message in order to inform and update other agents. The TGD, catching event messages from clusters and agents, sends update messages only to registered agents. During the stream of registration and deregistration events, the TGD keeps track of previously recognized simple services; this technique provides a certain amount of stability in the system to counteract variability in the sensor network topology.

4.1 Performance results

For our analysis of the TGD interaction, we have used the service registrations and deregistrations of the WSN simulation. The types of simple services amount to 31% of the total number of simple services, meaning that each simple service is about 3 times available in the network.
After the simulated 60 seconds, 97.5% of all complex services have been recognized by the TGD to be available in the network. Thus even if in our simulation the WSN resprects packet loss, the average number of available complex services is close to the maximum number of possible complex services considered in this experiment. Figure 5 analyzes the standard deviation of complex services registered at the TGD. We can see that in the first part of the simulation the TGD discovers complex services with high variability (the standard deviation reaches value of $\sigma = 9.2$), while at the end of the simulation the number of complex services over the simulation runs converge, showing the stability of our approach in provisioning complex services. After 20 seconds, complex services increase at a higher rate. This observation results from the fact that most of the complex services made of a few simple services can be found easier than others. At the end of the simulation, the TGD has discover 92% of all complex services with a low standard deviation of $\sigma = 1.5$.

5. FUTURE WORK

Future work will study algorithms evaluating optimal solutions for the current sensor network topology and to formalize a language for task graph description which considers the special requirements of service graph execution in Wireless Sensor Networks.

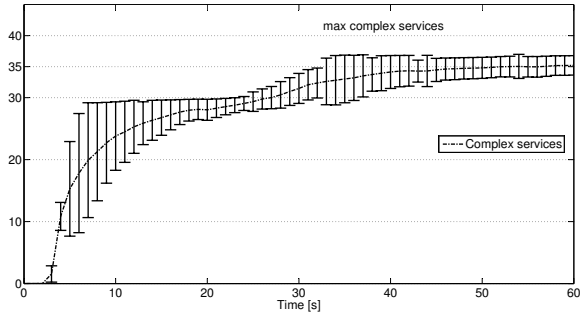

Figure 5: Results achieved by Service Discovery and TGD simulations, the avr. number of complex services discovered and the avr. standard deviation

6. ACKNOWLEDGMENTS

This work has been partly sponsered by the e-SENSE and SENSEI projects (www.ist-e-SENSE.org, www.sensei-project.eu) with the contract number IST-FP6-027227 and IST-FP7-215923. The authors would like to thank Roman Amstutz for the configuration the TOSSIM simulation and evaluation framework.

7. REFERENCES

[1] O. Amft, C. Lombriser, T. Stiefmeier, and G. Tröster. Recognition of user activity sequences using distributed event detection. In *2nd European Conference on Smart Sensing and Context*, 2007.

[2] P. Basu, W. Ke, and T. Little. A novel approach for execution of distributed tasks on mobile ad hoc networks. *IEEE Wireless Communications and Networking Conference*, 2, 2002.

[3] C. Brooks and E. Durfee. Congregating and market formation. *1. international joint conference on Autonomous agents and multiagent systems*, 2002.

[4] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communication & Mobile Computing*, 2(5), 2002.

[5] C. Campo, C. García-Rubio, A. López, and F. Almenárez. PDP: A lightweight discovery protocol for local-scope interactions in wireless ad hoc networks. *Computer Networks*, 50(17), 2006.

[6] R. Kumar, M. Wolenetz, B. Agarwalla, J. Shin, P. Hutto, A. Paul, and U. Ramachandran. DFuse: A Framework for Distributed Data Fusion. In *1. International Conference on Embedded Networked Sensor Systems*, 2003.

[7] M. Kushwaha, I. Amundson, X. Koutsoukos, S. Neema, and J. Sztipanovits. OASiS: A Programming Framework for Service-Oriented Sensor Networks. In *2. International Conference on Communication Systems Software and Middleware*, 2007.

[8] C. Lombriser, M. Stäger, D. Roggen, and G. Tröster. Titan: A Tiny Task Network for Dynamically Reconfigurable Heterogeneous Sensor Networks. In *15. Fachtagung Kommunikation in Verteilten Systemen*, 2007.

[9] W. Schott, A. Gluhak, M. Presser, U. Hunkeler, and R. Tafazolli. e-SENSE Protocol Stack Architecture for Wireless Sensor Networks. *16. IST Mobile and Wireless Communications Summit*, 2007.

[10] S. Seno, R. Budiarto, and T. Wan. Survey and new Approach in Service Discovery and Advertisement for Mobile Ad hoc Networks. *International Journal of Computer Science and Network Security*, 7(2), 2007.