Organizing Context Information Processing in Dynamic Wireless Sensor Networks

Clemens Lombriser [1], Mihai Marin-Perianu [2], Raluca Marin-Perianu [2],
Daniel Roggen [1], Paul Havinga [2], Gerhard Tröster [1]
[1] *Wearable Computing Laboratory, ETH Zürich, Switzerland*
{*lombriser, droggen, troester*}*@ife.ee.ethz.ch*
[2] *Pervasive Systems Group, University of Twente, The Netherlands*
{*m.marinperianu, r.s.marinperianu, p.j.m.havinga*}*@utwente.nl*

Abstract

This paper presents the e-SENSE middleware architecture for distributed processing of context information in dynamic wireless sensor networks. At the lower layer, the sensor nodes organize into clusters spontaneously, based on a shared context. These clusters form the basis for the service-oriented processing layer, where the functionality of the sensor network is expressed as service task graphs that support the distributed execution of applications. The higher layer is responsible for complex context inference and recognition. As a concrete example we evaluate the distributed recognition of human activities in a car assembly process.

1. Introduction

Ubiquitous and pervasive computing aims at integrating ambient intelligence in the everyday environment. Sensors embedded into objects and clothes can analyze their surrounding and derive context information, with the final goal of supporting people in their daily activities or at work. In a future networked world [4], the problem of organizing context information processing among the multitude of wireless smart objects becomes essential. The main challenges to be faced are: 1) the limited processing, low sensing quality, and scarce energy resources on the tiny integrated sensor nodes, 2) the selection of the sensors that can provide useful information, out of all available, 3) the dynamically changing environment due to mobility or other factors that perturb the wireless medium conditions.

This paper presents the approach of the IST e-SENSE project [12] in addressing these challenges. We describe the general architecture for the middleware developed and go into further detail on how to cluster sensor nodes to address the mobility of people and objects, how adaptive distributed processing is organized, and finally demonstrate the computation of context information in terms of user activity.

In order to improve the reliability of distributed processing in dynamic sensor networks, the sensor nodes need to be clustered such that the processing can be performed inside a group of sensor nodes which provide a certain stability of their communication links. For this purpose, we propose to use a context dependent measure, such as sensors recognizing whether they move together. The resulting clustering algorithm is presented in section 3.

The execution model for applications in the e-SENSE system is to interconnect services present on different sensor nodes to a service task graph. This service task graph is then dynamically executed in a distributed fashion on the sensor network as described in section 4.

The goal of e-SENSE is to compute context information from the wireless sensor networks. As an example of how this can be done, we discuss a distributed algorithm for the recognition of human activities in section 5. The algorithm works using simple detectors on all participating sensor nodes, which compute the activity as they perceive it. The classifications are collected at a central fusion node, which determines the actually performed activity, and can improve the overall accuracy of the recognition.

Section 6 summarizes the approaches and concludes the paper.

2. e-SENSE Middleware Architecture

The e-SENSE project aims at capturing ambient intelligence for beyond 3G communication systems through wireless sensor networks. For this purpose, it integrates wireless sensor networks into existing telecommunication networks and service infrastructures by the usage of gateway nodes, which connect various environmental and body sensor networks to the e-SENSE system. Context information queries that are posed to the e-SENSE system are computed in a distributed manner combining sensor readings from the different connected sensor networks.

The components of the e-SENSE reference protocol stack relevant for this paper are shown in figure 2. The protocol stack is divided into four logical subsystems; connectivity, middleware, management, and application. Each of the subsystems provide a variety of components and protocol entities, which allow optimizing for the individual sensor networks. We focus on the components relevant for this paper and refer the interested reader to [5] for a full description of the e-SENSE reference protocol stack.

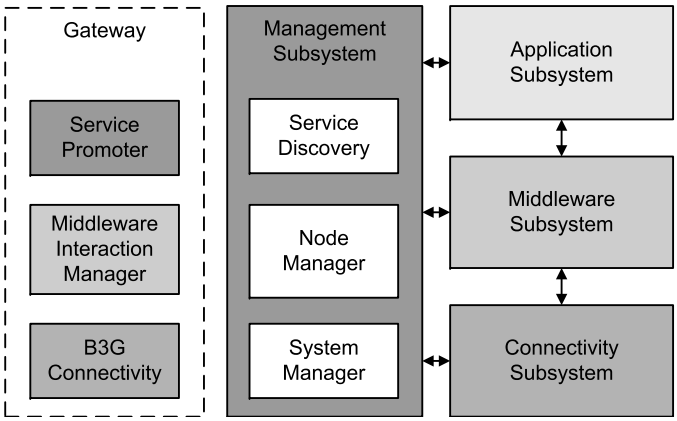

Fig. 1: e-SENSE protocol stack architecture with gateway extensions.

The *connectivity sublayer* of the e-SENSE protocol stack provides basic communication functionality. It includes node-to-node data transfer as well as network maintenance functions. The *middleware subsystem* builds on top of this functionality to create a network infrastructure for sensing, distributed processing, and connection over the gateways to the backbone network. The *application subsystem* hosts the applications programmed onto the sensor nodes, which can register their functionalities as services with the management subsystem.

The *management subsystem* is responsible for the configuration of different components and protocol entities in the other subsystem, such that a consistent and stable network is created. It contains a *Service Directory* protocol entity, which clusters the nodes in the network as described in section 3. These clusters represent cells of the network, in which processing is contained as far as possible. Every cluster elects a clusterhead, which instantiates a service directory and a System Manager. The service directory collects information on the services presented to the cluster by the different applications on the individual sensor nodes. The System Manager maintains the network configuration inside the cluster and organizes communication to clusterheads of neighboring clusters. Together with the Node Manager it organizes and maintains distributed processing algorithms as described in section 4.

On gateway nodes, the e-SENSE system adds three components that provide the interface between a B3G backbone network and the wireless sensor network. The Service Promoter translates and exchanges service information with the the e-SENSE backbone connected by the B3G system, such that services available in the sensor network can be used from other networks. The Middleware Interaction Manager is responsible for translating incoming and outgoing data formats between the two worlds. Those interfacing components allow the e-SENSE system to use data and service representations that are specific to a sensor network and allow considerable compression of the representation while keeping the flexibility in interconnecting different sensor networks.

3. Clustering and Service Discovery

The middleware subsystem provides the means for having multiple self-organizing wireless sensor nodes attached to mobile objects and persons, which are able to sense and distributively process contextual information. In order to achieve distributed processing in dynamic environments, we propose to organize the nodes that experience infrequent topological changes relative to one another into separate clusters. For example, a body area network can constitute a cluster, where all the processing needed for activity recognition is kept among the cluster members. Sensor nodes are then clustered when they are sharing the same context.

A special challenge in using context information for clustering is dealing with the accuracy provided by the context-sharing detection algorithms [7], [10]. Such an algorithm consists of an on-line calculation of the confidence that two nodes share the same context (for example whether they are moving together). The confidence value varies in time, which may cause undesireable fluctuations of the clustering structure. Therefore, the clustering algorithm has been designed with regard to the following requirements:

- **Dynamics:** The clusters can merge or split, depending on the context changes. Nodes can join and leave the cluster at any time if the topology or context changes accordingly. Contextual and topological changes cannot be predicted, so the clustering algorithm does not assume a stable situation during cluster formation.
- **Stability:** In order to provide a good basis for processing, the clustering structure has to be stable in case there are no contextual or topological changes. Therefore, the variations of the confidence values for sharing a common context must not lead to cluster instability. For this purpose, the decision to enter or leave the cluster is based on the history of context evaluations.
- **Leader election:** A leader for every cluster is dynamically elected in the process. The leader connects to the leaders of the neighboring clusters to form a communication backbone reaching over all clusters to the e-SENSE gateway.
- **Energy-efficiency:** As sensor nodes are typically restricted in their energy resources, the communication overhead should be kept to a minimum to prolong network lifetime.

The following section explains the clustering algorithm used by the e-SENSE system in more detail, and is followed by an evaluation of the performance of the algorithm.

A. Cluster Formation Algorithm

Following the requirements presented above, one node from each cluster is elected as *clusterhead*. In order to allow merging of clusters and to facilitate the election process, the candidate clusterheads dynamically generate unique *priority numbers*, either based on the unique hardware addresses, or as a context-dependant measure, such as the rapidity in occupying the wireless medium. A regular node subscribes to

the clusterhead with which it shares a common context and has the highest priority number.

Each node v periodically computes the confidence of sharing the same context with its neighbours. If the confidence with a neighbour u exceeds a certain threshold, then v considers that it shares the same context with u for the given time step. The final decision for sharing the same context with u is founded on the confidence values from a number of previous time steps, called the *time history*.

The algorithm constructs a set of one-hop clusters, based on the context information shared by the nodes. A node v can be: (1) *unassigned*, where v is not part of any cluster, (2) *root*, where v is clusterhead, or (3) *assigned*, where v is assigned to a cluster where the root node is one of its neighbours.

Any arbitrary node v in the network changes or chooses its root at every time step in the following cases: (1) v is unassigned, (2) v does not share a common context with its root, (3) the root of v is no longer a root and (4) v is root and there is another neighbour root, sharing the same context with v, that has a higher priority number. In one of these cases, v chooses as root node the neighbour root u with which it shares a common context and which has the highest priority number. If such a neighbour does not exist, v competes for clusterhead or becomes unassigned. The decision is based on the current status of the neighbours and tries to minimize the effect of the following erroneous situation: due to context fluctuations, an assigned node v may loose its root node and cannot join another cluster because none of its neighbours is root. Therefore, v may become root, form a new cluster and attract other nodes in that cluster. To avoid this undesirable outcome, a node declares itself root only if all its neighbours with which it shares a common context are unassigned. If there exists at least one neighbour u with which v shares a common context and u has a valid root node, then v becomes unassigned.

B. Cluster Stability Analysis

The evaluation of the clustering algorithm bases on a simulation of nodes attached to walking human bodies. The context recognition algorithm decides whether two sensor nodes are attached to the same body. We denote p as the probability of the correct detection of the *common* context and q as the probability of the correct detection of *different* contexts.

We simulate a network of 20 nodes, grouped in two clusters and we vary the probabilities p and q from 0.8 to 1. Figure 2 shows the performance of the clustering algorithm depending on the two probabilities which correspond to the accuracy of the context inference algorithm. We notice that in order to achieve stability of more than 0.9, the two probabilities have to exceed 0.98. As an example, the probabilities computed from the results reported by Lester et al. [7], who propose a context detection algorithm to deduce if two devices are worn by the same person, are $p \simeq q \simeq 0.96$, which gives a cluster stability of 0.64. We are now interested in how these probabilities change if we involve the time history in the decision process. The probabilities p_h and q_h of the correct detection of common context for a minimum time history h_{min} out of a total of H time steps is given by the CDF of the binomial distribution:

$$p_h(h_{min}, H) = \sum_{k=h_{min}}^{H} \binom{H}{k} p^k (1-p)^{H-k}, \quad (1)$$

$$q_h(h_{min}, H) = \sum_{k=h_{min}}^{H} \binom{H}{k} q^k (1-q)^{H-k} \quad (2)$$

If we take $h = 3,\ H = 5$, than we have $p \simeq q \simeq 0.99$ and a cluster stability of 0.96.

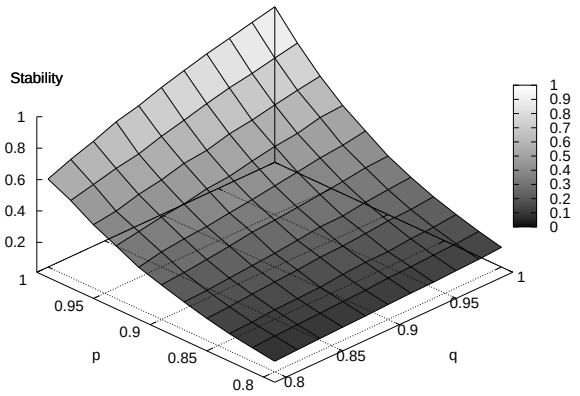

Fig. 2: Cluster stability depending on the probability p of correct detection of *common* context and the probability q of correct detection of *different* contexts.

C. Service discovery

The role of the leader nodes (or clusterheads) is to keep a directory of service registrations for the nodes in their clusters. The service directory is used for distributing the processing inside the cluster (see Section 4). Therefore, at the moment a node joins a cluster, it sends the description of the services it provides to the clusterhead node. An *internal* service discovery message is addressed to the clusterhead node, in order to check for a match in the registry corresponding to the desired service. An *external* service discovery message travels from one cluster to another, in search for a service that has not been found locally [11]. The two types of service discovery messages allow on one hand to quickly assess and access the capabilities of a cluster, and on the other hand to search for external services in case where multiple clusters are connected only for a short time.

4. Distributed Processing

The execution model of the e-SENSE system is based on the instantiation of services on the participating sensor nodes. A query for context information to the sensor network is translated into the invocation of a service. Such a service might either consist of a single simple service, which can be executed directly on one of the participating nodes, or might be composed of multiple services and a set of instructions on

how they need to be interconnected to perform the required complex service answering the context query. A complex service can thus be interpreted as a task graph, where the different simple services represent tasks. Using task graphs for distributed execution in Wireless Sensor Networks has been used before for data fusion [6], macroprogramming abstraction [3], or context recognition [8].

Upon a request for the retrieval of context information, the e-SENSE system translates the request into a task graph composed of services registered in the system. The composition of services might be influenced by the required computational quality, stability, or the services the system provides right now. The resulting task graph ready for execution is then sent by the e-SENSE gateway to the Middleware Interaction Manager (MIM), which stores it into a Task Graph Database as shown in Figure 3. The MIM then lets the System Manager start the execution of the task graph inside the network.

The services available at sensor nodes are stored in the local Node Manager. Applications on the sensor nodes can register services they want to provide to the e-SENSE system at this point. The implementation of the services are programmed into the sensor network nodes, and are only instantiated when they are needed. Services thus only occupy flash memory and get assigned a share of dynamic memory only when they are executed. This way the more scarce RAM resources are saved and more tasks can be made available on the sensor nodes. In a heterogeneous network, not every sensor node has the same processing capabilities. Only a selection of all possible tasks is thus stored in a *Service Pool* on every sensor node. Thus, nodes with high processing power can provide more and more complex tasks than simpler ones.

For processing, the System Manager distributes the services according to the capabilities of the sensor nodes and the requirements specified by service parameters. The *Task Graph Database* contains the service task graph descriptions to execute the required service. In the example in Figure 3, the Task Graph Database containins a context recognition service composed of sensor services S_i, feature services F_i, and a classification service C. Upon request to execute the algorithm, the System Manager inspects the currently available nodes in the network, and decides on which node to instantiate what services, such as to minimize processing load, overall power consumption, or network lifetime [6]. A weighting function is used for this purpose and favors nodes of the local cluster nodes to improve stability. The System Manager then sends a configuration message to the Node Managers on the sensor nodes, which instantiate the tasks on the local node. The System Manager assigns a share of dynamic memory to the tasks for their state information and configures the connections between tasks, including transmitting data to other nodes. As a consequence, a task is not aware of having been instantiated multiple times, nor that its surrounding tasks are not executed on the same node.

During the execution of the service task graph, the System Manager adapts the processing to eventual changes in the network topology, which might incur due to node movement

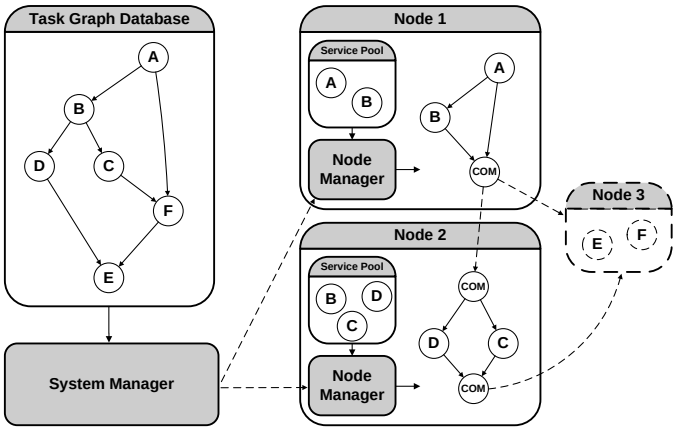

Fig. 3: Titan configures an application task graph by assigning parts of the graph to participating sensor nodes depending on their processing capabilities

or failure. In such a case, only nodes that experience a change in their configuration need to be reconfiguring, saving the state information in the tasks on the other nodes. Changes include replacing nodes, which are not available any more, or rearranging the task distribution for a more optimal solution according to the metrics used.

The results produced by the execution of the services in the e-SENSE system is returned via the Middleware Subsystem to the e-SENSE gateway. The Service Promoter translates the WSN internal data types back to a semantic description of the returned data, which is then delivered to the query issuer to respond his request.

The advantages of using the proposed architecture are:

- **Ease of use** – A designer of a context recognition algorithm can describe his algorithm simply by interconnecting different tasks and selecting a few configuration parameters for those tasks.
- **Portability** – Due to the abstraction of tasks, the framework is able to run on heterogeneous networks. The abstraction provides a good trade-off [8] between a fast custom application implementation, and adaptability to a range of different algorithm implementations on heterogeneous.
- **Flexibility** – As changes occur in a dynamic sensor network, the System Manager adapts the execution to more optimally use the given resources.
- **Speed** – Due to a compact configuration format and preprogrammed tasks, the Node Manager can reconfigure a sensor node in less than 1 ms [8].

5. CONTEXT RECOGNITION

Context recognition is an essential task of the e-SENSE system. In this section, we present a concrete application of recognizing human activities by using wireless sensor nodes worn on the body and integrated into working tools. The final goal is to provide assistance to workers in industrial environments, such as at a car manufacturing site. During the car assembly and test process, the workers perform various activities, such as “Mount the front door”, “Test the hood”,

TABLE 1: THE COMPLEX ACTIVITY "MOUNT THE FRONT DOOR" CONSISTS OF 7 BASIC OPERATIONS SENSED BY DIFFERENT DETECTOR NODES (PLACED AS INDICATED IN THE RIGHT COLUMN)

Basic activity	Detector locations
Pick up the front door	Right arm, Left arm, Front door
Attach the front door	Front door
Mount the screws	Right arm, Upper back
Pick up the socket wrench	Right arm, Upper back, S. wrench
Use the socket wrench	Right arm, Socket wrench
Return the socket wrench	Socket wrench
Close the front door	Left arm

etc. A system that recognizes these activities can support the workers with context-aware services, such as to present them detailed information about the current steps, or can automatically follow a checklist to ensure that all steps are correctly completed [14].

There are several difficulties in building a WSN-based activity recognition system: the inaccurate and noisy sensor data, the very limited resources (processing, memory and energy), the high variability of the data (e.g. for sensors placed on different parts of the body) and the unreliable, low bandwidth wireless communication. To overcome these problems, we propose a multi-layer, distributed activity recognition system. The recognition chain starts with the *detector* nodes, which are able to identify simple *events* from a continuous stream of acceleration data. A single event from one detector gives just an estimation of the real activity going on as it is perceived by the sensor at this location. However, fusing the information from several detectors reporting similar (or correlating) events within the same timeframe leads to a high confidence in recognizing that the user is performing a certain *basic operation*. For example, the basic operation "Pick-up the front door" can be inferred from the events signaled by the sensors on the user's body ("Pick-up something") and on the front door ("Front door picked-up"). Sequences of basic operations form the *complex activities*, which represent the final output of the recognition system. As an example, Table 1 shows the basic operations involved in the complex activity "Mount the front door", along with the detectors that contribute to recognizing these operations. We mention that in this example the order of the operations is an important parameter for recognizing and validating the complex activity.

A. Prototyping

The main tasks of the detectors are: 1) to spot relevant time frames in a continuous sensor data stream, and 2) to classify those correctly to one of the events to be reported by that detector. For this purpose, the detectors implement a feature similarity search (FSS) algorithm [1]. In the first phase, the data is segmented into windows with a step size of 250ms. In the second phase, the algorithm extracts a detector-dependent number of features from the data segments, in order to form the feature vector. The similarity of this feature vector to a previously learned set determines the classification decision. The similarity is measured by the Euclidean distance between the vectors.

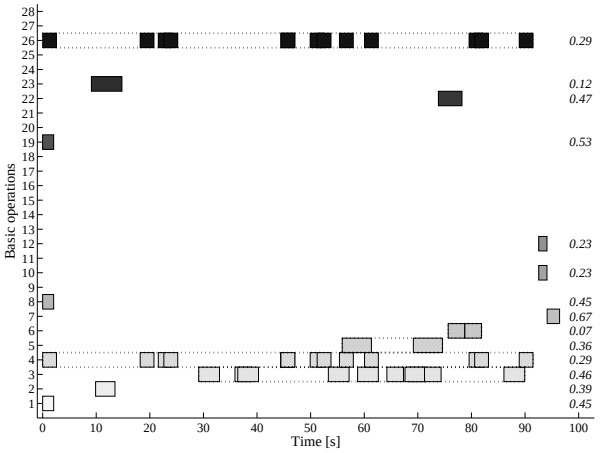

Fig. 4: Basic operations as reported by the detectors over time during the performance of the activity "Mount the front door". All operations above 7 are false positives, which can confuse the activity recognition. The temporal order is analyzed by combining all reports of basic operations (dotted lines) and checking for their correct sequence.

Two levels of data fusion are used to combine events into basic operations and subsequently to recognize the sequences of operations as complex activities. Figure 4 depicts one experimental data set collected when performing the activity "Mount the front door". We can identify two major difficulties. First, a low accuracy of the detectors may result into reporting false events (also referred to as *insertions*) and missing events (also referred to as *deletions*). Second, there might be overlaps among both the basic operations and the complex activities (i.e. the same events are involved in different operations and the same operations are performed during different activities), which can lead eventually to a misclassification due to confusion.

As a solution to these problems, we use fuzzy logic for performing the data fusion. Fuzzy inference systems (FIS) constitute an effective tool for wireless sensor networks, as they 1) can be implemented on limited hardware, and 2) can handle unreliable and imprecise information for a robust decision fusion under uncertainty [13]. The design of the FIS has to be adapted to the limited processing capabilities of the sensor nodes [9]. The confidence of the events reported by the detectors represent the FIS inputs. They are fuzzified using trapezoidal membership functions, for computational simplicity. The inference rules are derived directly from the description of the complex activities (for example, the rule corresponding to the activity "Mount the front door" has as terms the 7 operations listed in Table 1). Also for computational simplicity, we use sum-product inference and largest-of-maximum defuzzification.

An important extension is to include temporal order knowledge about the sequences of operations into the fuzzy inference. For this purpose, we add to the initial FIS an input variable for each activity class, derived from the temporal order. These variables characterize how well the sequences

of operations fit the activity descriptions, in other words they act as penalty functions for the operations appearing in the wrong order.

B. Evaluation

We evaluate the recognition system using an experimental data set [2] obtained from performing 9 car-assembly activities, each activity repeated 10 times by each of the 2 subjects. This resulted in a total of 180 experimental input data sets. The accuracy of the recognition system is evaluated using four-fold cross validation of training and validation data. The average classification accuracy is 90.14% for the normal FIS, and 93.53% for the temporal order extension. The system proves to be robust to deletions (caused by detectors missing events or occasional packet losses) and shows a good handling of insertions.

C. Integration into the e-SENSE framework

The distributed context recognition algorithm presented above can be integrated into the e-SENSE framework by splitting the feature calculations and FSS of the detectors and basic operation inference as well as the activity detection of the fusion node into a set of services. To compute the context information, an overall service task graph is defined using these services. This service task graph can be reused for different scenarios by parameterizing the services in a different way. Using a multitude of service task graphs, the context algorithm can be adapted to the available services as well as the required quality of service.

6. CONCLUSION

The e-SENSE project aims at capturing ambient intelligence for beyond 3G communication systems through wireless sensor networks. We have presented the approaches it takes to enable distributed context inference inside the wireless sensor networks connected to the e-SENSE system. In a first step, the sensor nodes are clustered according their context to form a stable basis with little topological changes for distributed processing. In a second part, the processing is organized in those clusters by building a service task graph from services available on the participating sensor nodes and assigning each node a subset of the service task graph for execution.

How context information can be inferred in a distributed way using wireless sensor networks has been shown in an example of recognizing human activities in a car assembly scenario. Each sensor node detects the activities individually, as far as it can perceive them. Subsequently, a distributed fuzzy inference system fuses these individual observations, in order to derive a reliable aggregate decision of the actual activity performed.

ACKNOWLEDGMENTS

This paper describes work undertaken in the context of the e-SENSE project, ‘Capturing Ambient Intelligence for Mobile Communications through Wireless Sensor Networks’ (www.ist-e-SENSE.org). e-SENSE is an Integrated Project (IP) supported by the European 6th Framework Programme, contract number: 027227.

The authors would like to acknowledge the contributions of their colleagues from the e-SENSE consortium.

REFERENCES

[1] O. Amft, H. Junker, and G. Tröster. Detection of eating and drinking arm gestures using intertial body-worn sensors. In *9th International Symposium on Wearable Computers (ISWC)*, pages 160–163, 2005.

[2] 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 (EuroSSC)*, 2007.

[3] A. Bakshi, V. K. Prasanna, J. Reich, and D. Larner. The Abstract Task Graph: a methodology for architecture-independent programming of networked sensor systems. In *Workshop on End-to-end, sense-and-respond systems, applications and services*, pages 19–24, 2005.

[4] H. Gellersen, A. Schmidt, and M. Beigl. Multi-Sensor Context-Awareness in Mobile Devices and Smart Artifacts. *Mobile Networks and Applications*, 7(5):341–351, 2002.

[5] A. Gluhak. e-SENSE Architecture. In *3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2007.

[6] R. Kumar, M. Wolenetz, B. Agarwalla, J. Shin, P. Hutto, A. Paul, and U. Ramachandran. DFuse: A Framework for Distributed Data Fusion. In *1st International Conference on Embedded Networked Sensor Systems (SenSys)*, pages 114–125, 2003.

[7] J. Lester, B. Hannaford, and G. Borriello. “Are You with Me?” – Using Accelerometers to Determine If Two Devices Are Carried by the Same Person. In *2nd International Conference on Pervasive Computing (Pervasive)*, pages 33–50, 2004.

[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 (KiVS)*, pages 127–138, 2007.

[9] M. Marin-Perianu and P. J. M.Havinga. D-FLER: A distributed fuzzy logic engine for rule-based wireless sensor networks. In *International Symposium on Ubiquitous Computing Systems (UCS)*, 2007.

[10] R. S. Marin-Perianu, M. Marin-Perianu, P. Havinga, and J. Scholten. Movement-based group awareness with wireless sensor networks. In *5th International Conference on Pervasive Computing (Pervasive)*, pages 298–315, 2007.

[11] R. S. Marin-Perianu, J. Scholten, P. J. M. Havinga, and P. H. Hartel. Energy-efficient cluster-based service discovery in wireless sensor networks. In *31st IEEE Conference on Local Computer Networks*, pages 931–938, November 2006.

[12] M. Presser, A. Gluhak, D. Babb, L. Herault, and R. Tafazolli. e-SENSE Capturing Ambient Intelligence for Mobile Communications through Wireless Sensor Networks. In *15th IST Summit Myconos*, `http://www.ist-esense.org`, 2007.

[13] V. Samarasooriya and P. Varshney. A fuzzy modeling approach to decision fusion under uncertainty. *Fuzzy Sets and Systems*, 114(1):59–69, 2000.

[14] T. Stiefmeier and C. Lombriser and D. Roggen and H. Junker and G. Ogris and G. Tröster. Event-Based Activity Tracking in Work Environments. In *3rd International Forum on Applied Wearable Computing (IFAWC)*, 2006.