

# DarSens: A Framework for Distributed Activity Recognition from Body-Worn Sensors\*

Michael Haslgrübler  
Johannes Kepler University Linz, Institute for  
Pervasive Computing  
Altenberger Straße 69, 4040 Linz, Austria  
michael@haslgruebler.eu

Clemens Holzmann  
Upper Austria University of Applied Sciences,  
Mobile Computing  
Softwarepark 11, 4232 Hagenberg, Austria  
clemens.holzmann@fh-hagenberg.at

## ABSTRACT

With the increasing amount of sensors in our environment, the desire to reuse existing sensors for different applications grows. However, most appliances do not provide access to their sensors in a cross-application manner, but rather use them for a specific purpose only. In this paper, we describe a framework which provides a way to access not only the data, but also the processing capabilities of a sensor system in a reusable way, without the need for a-priori knowledge about the availability of sensors in the environment. In particular, the presented framework is able to run on an embedded system platform, and is used for the recognition of human activities in a body sensor network. Notably, both the feature extraction and classification are performed within the network. Hence, we can use the processing power of sensor nodes, and do not have to revert to the processing capabilities of a client device which is using the sensor network. Two experiments have been conducted to show the feasibility and performance of our approach in typical activity recognition scenarios.

## Categories and Subject Descriptors

C.2.4 [Computer-Communication Networks]: Distributed Systems, Network Protocols

## Keywords

Body sensor network, context-awareness, activity recognition, self-organization

## 1. INTRODUCTION

In order to create a context-aware system which requires less human intervention, it is necessary to recognize the situation of the user – or more precisely, the activity he is currently performing – as well as the context of his surrounding,

\*This work is supported by the FP7 ICT Future Enabling Technologies programme of the European Commission under grant agreement No 225938 (OPPORTUNITY).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

BodyNets '10 September 10-12, 2010, Corfu Island, Greece  
Copyright 2010 978-963-9799-41-7

and adapt to them accordingly. Due to the recent advances in sensor technology and low power wireless communication (e.g. IEEE 802.15.4), it has become possible to create sophisticated applications that are context-aware. Typical application scenarios of previous research work which include activity recognition from on-body sensors are physical rehabilitation [5], living assistance [13] – i.e. calling help when an emergency is detected – and HCI [9], among others.

Most activity recognition systems work with acceleration sensors and apply feature extraction and supervised machine learning techniques to acquire the information in need [1]. However, in most cases the systems are tailored to a certain amount of sensors and a specific deployment scenario. In order to advance further, we need to be more flexible in the way how sensors are integrated in context-aware systems. Therefore, we propose the reuse of sensors that are available in the environment, by enabling them to be accessed in a standardized way. In the following, we will present an approach for discovering and using wireless sensors in an ad-hoc manner. It makes best use of the available sensors, and provides cross-application access to the data and processing power of sensor nodes in an ad-hoc fashion. Our approach is a shift away from deployment scenarios with statically defined communication patterns, towards more dynamic and opportunistic sensor configurations. Furthermore, we present its implementation in a framework referred to as **DarSens**, which enables the recognition of human activities in a fully distributed fashion.

The main contributions of this paper are:

- Introduction of a framework for *distributed activity recognition*, which allows sensor acquisition, feature processing and classification within the network and without a centralized control.
- Presentation of a mechanism to *orchestrate sensors* into a structure that is suitable for a goal-oriented information exchange, which means that the sensors interact to achieve a common activity/context recognition goal. According to [12], the goal formulation says what should be recognized, but does not specify how. This allows the system to autonomously configure itself without the needed for a centralized control.
- Evaluation of the framework by measuring (i) the time needed for this orchestration, as well as (ii) its performance in terms of classification accuracy, power efficiency and communication effort while a user is performing different types of activities.

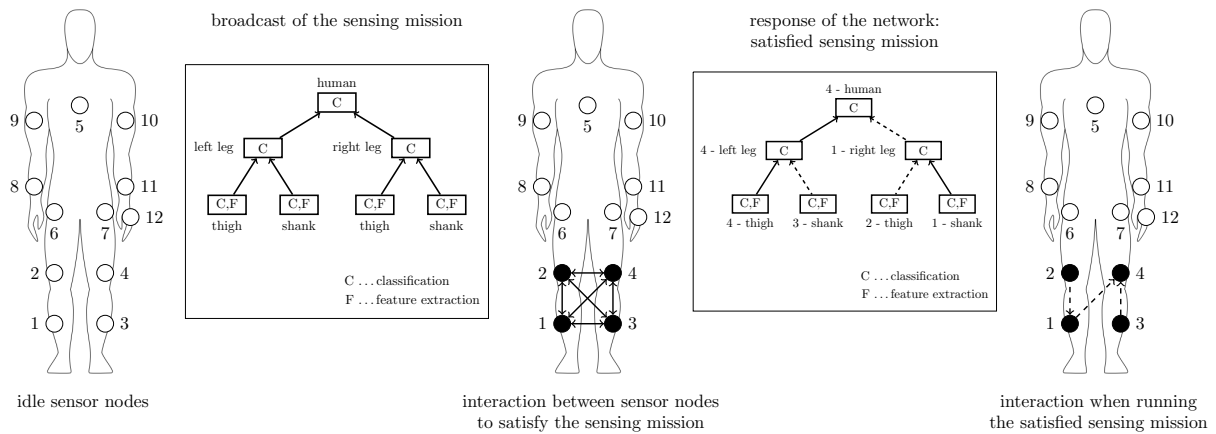


Figure 1: From sensors to a working sensor network ensemble.

## 2. OPPORTUNISTIC RECOGNITION OF HUMAN ACTIVITIES

This work is part of the OPPORTUNITY project [12, 11], which explores the opportunistic recognition of activities by making best use of the sensors available in an environment. In order to do so, three sequential steps have to be performed:

1. formulation of a *recognition goal*
2. transformation of the recognition goal into a coordinated *sensing mission*
3. communication of the sensing mission to the available sensor nodes

A *recognition goal* represents the activities a context-aware application is interested in. For example, a mobile device could be interested in locomotion activities of the user, in order to automatically accept or reject incoming calls; e.g., if the user is *running* to catch the train, he will not be able to pick up the phone, but he probably will be able to do so if he is *standing* and waiting for the train.

A *sensing mission* is a blueprint for the sensor network, specifying which capabilities are needed to fulfill the recognition goal, like for example which sensors of a body sensor network are required to recognize locomotion activities. The transformation of a recognition goal into a sensing mission should be done automatically, e.g. by looking up a database which sensing mission provides the required information.

Finally, the sensing mission is communicated to a wireless sensor network. Upon reception, the sensor nodes compare their own capabilities with those specified in the sensing mission, and participate – sometimes with the help of other sensors – by delivering the requested information. This participation and interaction with other nodes happens in an ad-hoc manner, without a-priori knowledge about the sensor network. For example, sensors attached to a human’s legs could cooperate to provide locomotion activities, as specified in the sensing mission, to a mobile phone.

## 3. REALIZATION IN DARSSENS

In order to perform opportunistic activity recognition in **DarSens**, the following steps are essential (cf. Figure 1):

1. An application (the requester) broadcasts a sensing mission (SM) to fulfill a certain recognition goal. It is organized as a tree data structure, which will be explained in detail in Section 3.1.
2. Sensors receive the sensing mission and *self-organize* to satisfy the respective information needs. If not all data needed for a sensing mission can be provided by the available sensors, the requester may have to choose another sensing mission.
3. Once the sensing mission can be satisfied, the sensor network responds with an annotated sensing mission that contains the MAC addresses of the participating sensors, therewith confirming that all data can be gathered and processed.
4. In order to start the recognition process, the requester approves the sensing mission by responding with a corresponding notification. This in turn starts the recognition process at the participating sensor nodes, which will start to exchange information and send the recognized activity information to the requester.

In the following sections, it will be explained in detail how this behavior is implemented and how it can be used for building own applications.

### 3.1 Containment Hierarchy

In **DarSens**, a sensing mission is represented by a tree data structure that defines (i) from which parts of the human body sensor data needs to be gathered, and (ii) how it has to be processed by the sensor nodes.

The hierarchy is based upon the anatomy of the human body, as can be seen in Figure 1. The root node is always referred to as *human*, and its child nodes represent the extremities; these are *head*, *left arm*, *right arm*, *torso*, *left leg* and *right leg*. Those child nodes may of course be further segmented, e.g. for *left leg* the child nodes would be *thigh*, *shank* and *foot*. With this structure, position information on a human body can be modeled. We assume that every sensor knows its own position in the tree, either pre-defined or by a localization approach such as those presented in [7]. The containment hierarchy, which represents the position of

a sensor, is stored locally on the sensor nodes and used by them for the purpose of self-organization.

The containment hierarchy is not only used to represent the position of a sensor, but it also defines the recognition chain, i.e. the information flow from input devices such as acceleration sensors towards an output device such as a mobile phone. At the leaf nodes, sensor data is being extracted – from the body parts on which they are placed – and fed into the recognition chain. At every node in the hierarchy, feature extraction or classification can be performed, which is specified in a sensing mission with 'F' and 'C', respectively (cf. Figure 1). The output of the performed tasks – data acquisition, feature extraction and/or classification – is the input for the next level in the hierarchy. The sensor nodes, which execute the tasks at the leaf nodes, also perform the tasks of their ancestors. As the leaf nodes of a sensing mission share their ancestors, multiple sensor nodes would be applicable to perform the tasks of the ancestors; however, only one sensor node will be performing those tasks. This selection process is part of the self-organization phase described later on.

In addition, the concept of a *satisfied sensing mission* is introduced. A satisfied sensing mission is a tree data structure which contains all the information of a sensing mission, and additionally every node is annotated with the MAC address of the sensor which will execute the tasks of this node; in Figure 1, the MAC addresses are symbolized with numbers from 1 to 12. They are needed because multiple sensor configurations may be able to satisfy a certain sensing mission, and therefore it is necessary to select which ensemble of sensors should deliver the requested information. For example, a sensing mission consisting of the *human* node only would be satisfiable by many sensor nodes, for which reason the requester of this sensing mission has to select one and discard the others.

It should be noted that instead of using the human anatomy for the containment hierarchy, other domain models could be used as well, like e.g. a building as the root node with the containing floors as its child nodes.

### 3.2 Self-Organization

A sensor node participates in a sensing mission if and only if its local containment hierarchy, representing its position on the human body, *overlaps* with a received sensing mission, and if its sensors are useful for the recognition process. The sensor data is only useful if a leaf node of the sensing mission – and therefore also the leaf's ancestor nodes – overlaps with the local containment hierarchy of a sensor. If this is not the case, the data of the sensor is not useful; for example, if sensor data from the left hip is needed, data from the left thigh is not useful (cf. Figure 2).

If the local tree and the sensing mission overlap, the sensor will create a satisfied sensing mission by inserting its MAC address at the overlapping nodes (cf. Figure 3(a)-3(c)), if and only if there is not already a MAC Address assigned. A sensor now has two ways to proceed: (i) if all the nodes have a MAC address assigned, the satisfied sensing mission is complete, and the sensor will send it to the requester; (ii) if not all nodes have a MAC address assigned, the sensor will create a *sub-sensing mission*, which is basically the difference tree between the local tree and the original sensing mission, and broadcast it (cf. Figure 3(d)). Another sensor may receive this request, satisfy the sub-sensing mission

and send the *satisfied sub-sensing mission* to the requester (cf. Figure 3(d)-3(f)). Now the satisfied sub-sensing mission will be merged with the rest of the existing satisfied sensing mission shown in Figure 3(c). The result – see Figure 3(g) – will eventually be sent to the original requester upon completion.

As sensors can create sub-sensing missions, it is possible to apply this approach in an environment where not every sensor can communicate with other sensors or the requester. For example, this may be the case if the transmission power of a sensor node is reduced for power-saving purposes. It still could participate in a respective sub-sensing mission, although it is too far away from the original requester of the sensing mission.

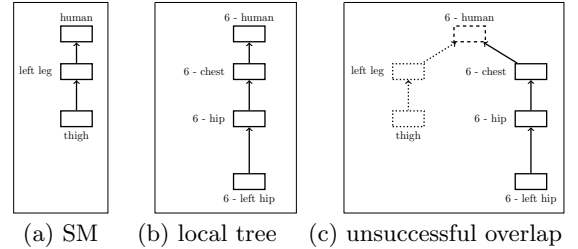


Figure 2: Participation in a sensing mission.

### 3.3 Node Set Up, Tear Down and Data Flow

After the self-organization of the network, the requester of a sensing mission may receive one or more satisfied sensing missions – e.g. because multiple sensor ensembles are applicable for the same sensing mission – and choose one of them (e.g. first-come first-served). A notification will be sent to the deliverer of the selected satisfied sensing mission, which is also its root node. Upon arrival of the notification, and based on the specification in the sensing mission, feature extraction and/or classification may be performed at the root node. If necessary, additional information – in our case, this is a list of features to extract and/or a classification model – will be fetched from the requester. Afterwards, it will relay the notification to the intermediate children in the containment hierarchy of the sensing mission, and they will perform the same procedure by examining the nodes to which they are assigned. This is done in parallel for children and sequentially per depth level of the hierarchy of the sensing mission.

When the setup procedure reaches the leaf nodes of the sensing mission, a check if feature extraction or classification are needed is performed, sampling of sensor data – which are accelerations in our work – will start, and the data will be fed into the feature extraction chain or the local classifier. Either the raw data or the inferred data (i.e. features or classification results) will be relayed to the ancestors for further processing. Ancestor nodes will receive the data from their children and may process it further or just relay it to the next level, which may be the requester or another ancestor node.

To stop executing a sensing mission, the requester will send a corresponding notification to the root node of the satisfied sensing mission, which will in turn relay this notification to its children.

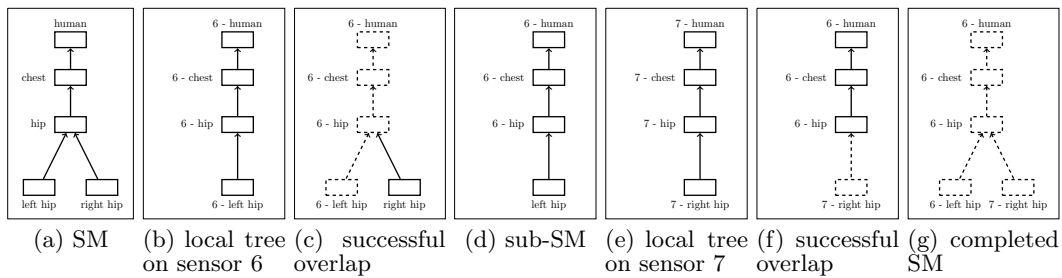


Figure 3: Exemplary self-organization process.

### 3.4 Self-Adaptation

In order to provide a robust system for continuous activity recognition, the framework will try to *restore from failures* – e.g. segmentation faults, battery failures or temporary transmission errors – autonomously. When a satisfied sensing mission is deployed to the sensor network and running, a sensor node will detect such a failure by receiving no more data from its child node in the hierarchy. Now, the sensor node has several options to proceed:

- (i) notify its children to continue delivering information
- (ii) create a sub-sensing mission for the failed part of the sensing mission
- (iii) notify its ancestor to terminate the satisfied sensing mission

First, the sensor node which detects the failure will try to notify the child node that failed to deliver the information by creating and sending a satisfied sub-sensing mission for the failed part of the network of which the child node is the root. If the node failed for a short period of time, it may have lost all previously known runtime information; in this case, it starts the set-up routine, and continues to deliver the information to its ancestor upon arrival of the satisfied sub-sensing mission.

If strategy (i) fails, the sensor node which detected the failure will create a sub-sensing mission for the failed part of the network and broadcast it. If there is another sensor node – or a group of sensor nodes – available that can fulfill this sub-sensing mission, they will be integrated into the overall sensing mission.

If strategy (ii) also fails, the error is propagated towards the ancestors of the node that detected the failure, which in turn notify their children to stop running the sensing mission. Eventually, the information arrives at the requester, which can decide how to proceed. For example, the requester could select another sensing mission, which does not contain the failed nodes.

Typically, the system will recover from most short-time failures using strategy (i). The time needed to recover is the time for the set-up procedure of the failed network part. In the case of strategy (ii), the time to recover comprises the time for the self-organization and the time for the setup procedure. In both cases, the time will be the less the deeper the failure occurred within the containment hierarchy of the sensing mission. Please note that in any case the recognition will not be stopped, but continued using the last received information of the failed network part. As for strategy (iii),

the time is the same as in strategy (ii), given that the failing node was the root node of the sensing mission. This is of course the worst case scenario, since the self-organization and the setup procedure have to be performed for the whole containment hierarchy of the sensing mission.

Through the combination of self-organization and self-adaptation, it is possible to cache and reuse previously received satisfied sensing missions to start the recognition process without a self-organization phase, even if some of the nodes assigned to satisfied sensing mission do not exist any longer. Those missing parts will be replaced later within the network by the self-adapting behavior; the only exception in this case is the root node of the sensing mission’s containment hierarchy, as it is the root node – i.e. the node which communicates with the mobile device – for the satisfied sensing mission.

### 3.5 Technology

**DarSens** is implemented on the Sun Microsystems Small Programmable Object Technology (SPOT) platform, cf. Figure 4(a), which contains a 180 MHz ARM CPU, 512KB RAM, 4MB Flash, an IEEE 802.15.4 RF module, a 3-axis accelerometer, 8 tri-color LEDs and a few GPIOs which can be used for extending the SPOT. Please note that **DarSens** builds upon the software framework of the Sun SPOT platform.

### 3.6 Workflow for Building Applications

In order to build an application using **DarSens**, it is necessary to create a sensing mission, define the classifier models and feature matrices for each node, and integrate the framework into an application. For obtaining the classifier models, a few steps are necessary. First, the activities that have to be distinguished must be defined, and also the body parts which take part in those activities. The next step is to record the raw acceleration data from the sensor placed on those body parts. Afterwards, the raw data can be post-processed to calculate all the features which are implemented by the framework. Finally, the data has to be labeled – a GUI tool for that purpose is included with the framework – and converted into an *arff* file, which can eventually be used by the WEKA toolkit [2] to obtain the classifier models.

The complete source code of the framework as well as a few sample applications can be downloaded at <http://darsens.sourceforge.net>.

## 4. EVALUATION

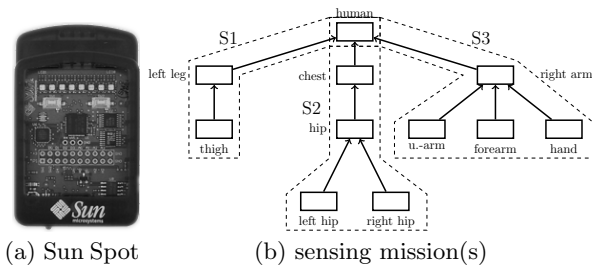
Two experiments have been conducted to show the feasibility of our approach and its implementation. First, we

measured the time needed for the sensors to self-organize and satisfy different sensing missions. Second, we evaluated how our system performs in a typical activity recognition scenario.

Since the system needs to self-organize in order to start the recognition process in the first place, the time needed for the self-organization should be reasonably fast, as shown in the first experiment. Please note that by caching and reusing previously obtained satisfied sensing missions, this time can be further minimized by an application developer for reoccurring sensing missions.

In any activity recognition application, there has to be a trade-off between certain requirements. We consider the following three requirements crucial for every such application: (i) accuracy, (ii) speed and (iii) energy efficiency. In the second experiment, we show how these requirements can be applied in **DarSens** by using different sensing missions, and how the system behaves under these constraints.

## 4.1 Self-Organization



**Figure 4: Reference platform and sensing mission(s) for self-organization experiments.**

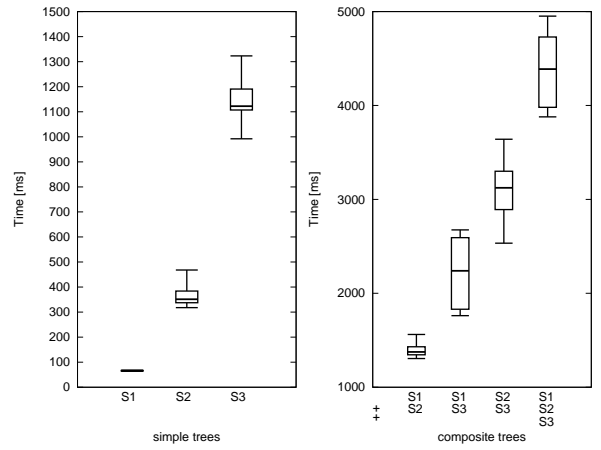
In the first experiment, we measured the time needed for the self-organization of the body sensor network. The measurement was started with the broadcast of a sensing mission at the requester, and stopped when the first satisfied sensing mission – additional satisfied sensing missions are discarded – arrived. We considered a set of sensing missions in our experiment, namely S1, S2, S3, and compositions of them (cf. Figure 4(b)). Each sensing mission was broadcast ten times, and the measurements are presented in Figure 5.

One clear trend is that the more nodes are participating in a sensing mission, the more time it takes to satisfy it. This is due to the nature of broadcasts, which we have to use as we do not know the available sensor nodes in advance. When using broadcasts, there is no reception guarantee, and hence any request may get lost and may have to be re-requested. For more details, please refer to our previous publication [4].

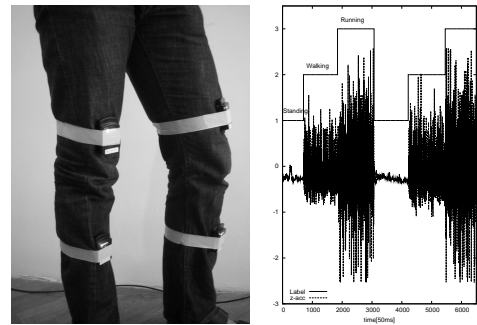
## 4.2 Activity Recognition

In order to evaluate our framework further and show its feasibility, we consider a typical activity recognition scenario. We want to distinguish three types of activities – running, walking and standing – and take a supervised learning approach.

For training the classifiers, a dataset of 6 minutes was recorded on four body positions (thigh and shank on both legs) and labeled with the three activities, whereas each activity was performed for one third of the duration of the recording; the deployment and the labeled sensor data are



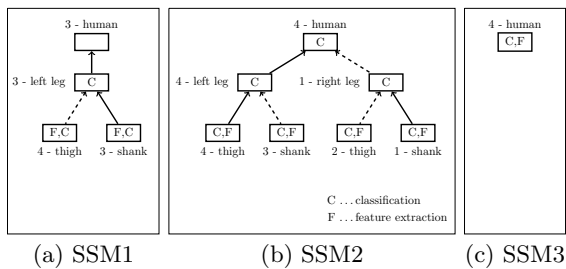
**Figure 5: Durations of the self-organization.**



**Figure 6: Deployment and labeled data.**

shown in Figure 6. The feature extraction was performed on the dataset – standard deviation, energy, mean, mean crossing rate, zero crossing rate, variance, root mean square and fluctuation, which have been used in previous research [8] also – and classification was performed using the WEKA machine learning toolkit [2]. We used the J48 classifier, as it is among the best classifiers for activity recognition according to [10], and because of the fact that its textual representation easily allows to send the classifier model to a sensor node and interpret it there at runtime. Since we can transfer models at runtime, we recorded and evaluated the data on the same person, as we can easily change our models without the need to flash the hardware, and hence a cross-person model is not necessary. To reduce the data set, only forward and upward acceleration were used (Y- and Z-axis), and additionally a classifier subset evaluation was performed to diminish the feature set further by best-first-search.

For training the classifiers of each sensing mission, we used only the training data gathered by the sensor nodes on the corresponding body positions. E.g. for SSM1 in Figure 7 only the recorded data of the left leg’s thigh and shank was used, more precisely the data recorded on the left leg’s thigh was used to train the classifier at the corresponding node in the sensing mission, in an analogous manner for the shank. As for training the classifier in SSM1’s *left leg* node the output of the trained classifiers of its child nodes *thigh* and *shank* were used.



**Figure 7: Exemplary satisfied sensing missions with marked wireless connections.**

We have chosen the three different sensing missions shown in Figure 7, which were deployed to the sensor network of Figure 6. Intuitively, the sensing missions would be labeled with our requirements as follows: SSM3 would be the most energy efficient and fastest, SSM2 the most accurate sensing mission and SSM1 would be a good compromise.

The performance of the sensor network was measured while performing the activities running, standing and walking for six minutes. In all sensing missions, the features are inputs for a classifier on the same node, which saves a lot of bandwidth because the data to be transmitted is reduced. The meta-classifiers of SSM1 and SSM2 use classification output of their descendants as inputs and classify accordingly.

In Figure 7, exemplary satisfied sensing missions are illustrated. The number symbolizes the MAC-address of a certain sensor node, and – as previously stated – shows that the tasks of more than one tree node are executed on the same sensor node. Therefore, the data flowing bottom-up does not have to be transmitted wireless all the time (see the dotted line in Figure 7), but will be transmitted locally (see the solid line in Figure 7). Furthermore, sensor/classification/feature extraction data that contains the same information as the previously gathered will not be transmitted every time to save bandwidth. However, sometimes an update of data is necessary to avoid that the ancestor node detects a failure by the absence of information, which would in turn start self-adaptation. This update of data doesn't have to be requested but will occur automatically before a respective ancestor node would start the self-adaptation behaviour.

Two accuracies of the activity recognition process are presented in Table 1. The first one was calculated by the WEKA Toolkit, while training the classifiers, and the second one is the accuracy of the sensor network output. Looking at the output of the system, it was noticeable that the accuracy dropped significantly for all sensing missions. This may have been caused by several factors like displacement, which can reduce the accuracy by 72% as shown in [6], and that the training data may have been too small.

The time to data describes the time-span for receiving the first activity information by the requester. There is a two seconds delay in fetching the classifier model as well as information about which features to extract. This is done sequentially per hierarchy level, but in parallel within each hierarchy level. Therefore, the amount of time needed for the first data to be received is higher if the containment hierarchy level is deeper, but only slightly higher if there are more nodes which can be seen in the difference between SSM1 and SSM3.

[8] has shown that the sampling frequency where activity recognition from acceleration data stabilizes is between 15Hz and 20Hz; therefore, our system uses 20 Hz not only as its sampling frequency, but also as its operating frequency.

To conclude, our intuition has been confirmed, namely that SSM2 is the most accurate, SSM3 the fastest and energy efficient sensing mission, and that SSM1 lies in-between. Interestingly, the amount of data transmitted scales well with the amount of nodes used in the sensing missions. We also experimented with a sensing mission similar to SSM1, but without local classification at the leaf nodes; however, the amount of data that had to be delivered was too much for the network to handle, and the sensor nodes were not able to continue their work after some time.

tree	SSM1	SSM2	SSM3
number of nodes	2	4	1
wireless connections	2	4	1
LIVE accuracy (%)	79	85	73
WEKA accuracy (%)	98.5	99.3	97.8
self-org. time (ms)	518	2395	75
time to data (ms)	2422	3132	2162
memory usage (KB) up to	350	363	309
comm (KB)	75	150	35
energy consumption (%)	1 – 2	1 – 2	1

**Table 1: Comparison of different sensing missions.**

## 5. RELATED WORK

Activity recognition systems have been investigated for a long time and in various application scenarios; a good overview of previous research can be found in [1].

Another related issue are service discovery protocols. In [14], service discovery is surveyed as a solution to the problems present in pervasive computing environments, which are highly dynamic and heterogeneous and need to incorporate computing devices and people. When dealing with systems that implement Jini-like service discovery mechanisms, one needs to look at underlying modalities in terms of service discovery, registration and infrastructure. The discovery process is either announcement-based – i.e. every participant listens to service announcement of others – or query-based – i.e. participants request service information from their environment. In terms of infrastructure, such systems either use a directory or a non-directory based approach. In the first case, a centralized authority holds information about the services available in the surroundings, and services will register their capabilities with this authority. In the latter case, every participant provides information about available services, the list of services is aggregated using the announced based discovery approach.

When using such systems in a pervasive environment, a few problems arise, as they are – in contrast to **DarSens** – not designed to work with unknown environments. When considering announcement-based discovery of services and non-directory based infrastructures, the network is flooded at a regular basis with such announcements, which is not suitable for battery powered devices, as wireless communication is one of the main power consumers. When considering the query-based approach without a centralized authority, it will need a lot of retries to find the required service. On

the other hand, the location of such an authority needs to be determined – if it exists in the present environment at all – before communication between the service user and provider can be established. Finally, if more than one service needs to be combined, this combination can only occur at the requester site. This is contrary to the approach within **DarSens**, where service and data composition take place locally, which reduces bandwidth throughout the network and therefore saves energy.

Like **DarSens**, more recent work also emphasized the use of multiple processing layers [3], separating the processing of signals, features and classification onto different components throughout the system. However, this approach differs insofar from ours as the separation of processing task is pre-defined based on the level within the system, without considering the structure of information in postures.

## 6. CONCLUSION AND FUTURE WORK

In this paper we presented **DarSens**, a framework for accessing sensor data and processing capabilities in a cross-application manner, which can be used to perform activity recognition in a distributed sensor system. The system uses – to the best of our knowledge – a novel approach for orchestrating sensors available in the environment into cooperative sensor ensembles. We have conducted two experiments: the first one evaluated the time needed for orchestrating sensors into a sensor ensemble, the second one measured the performance when using the system to perform activity recognition with and without multiple hierarchically linked classifiers.

While the latest results look already promising and show the feasibility of our work, we plan to further evaluate the system performance in various user scenarios. Furthermore, it would be interesting to perform a user study with developers, who use the system for the rapid prototyping of context-aware applications. We would also like to further ease the development of such applications with the framework and automate the work flow such that developers only need to label the sensor data gathered.

## 7. REFERENCES

- [1] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In *Proceedings of the 2nd Int'l Conference on Pervasive Computing, Pervasive 2004*, volume 3001 of *LNCS*, pages 1–17. Springer, Apr. 2004.
- [2] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. *The WEKA Data Mining Software: An Update*, volume 11. ACM, 2009.
- [3] H. Harms, O. Amft, G. Tröster, and D. Roggen. Smash: A distributed sensing and processing garment for the classification of upper body postures. In *Proceedings of the 3rd Int'l Conference on Body Area Networks, BodyNets'08*, pages 1–8. ICST, Mar. 2008.
- [4] C. Holzmann and M. Haslgrübler. A self-organizing approach to activity recognition with wireless sensors. In *Proceedings of the 4th Int'l Workshop on Self-Organizing Systems, IWSOS 2009*, ETH Zurich, Switzerland, December 2009. Springer LNCS.
- [5] E. Jovanov, A. Milenkovic, C. Otto, and P. de Groen. A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation. *Journal of NeuroEngineering and Rehabilitation*, 2(1):6, 2005.
- [6] K. Kunze and P. Lukowicz. Dealing with sensor displacement in motion-based onbody activity recognition systems. In *Proceedings of the 10th Int'l Conference on Ubiquitous computing, UbiComp 2008*, pages 20–29, New York, NY, USA, 2008. ACM.
- [7] K. S. Kunze, P. Lukowicz, H. Junker, and G. Tröster. Where am I: Recognizing on-body positions of wearable sensors. In *Proceedings of the 1st Int'l Conference on Location- and Context-Awareness, LoCA 2005*, volume 3479 of *LNCS*, pages 264–275. Springer, 2005.
- [8] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher. Activity Recognition and Monitoring Using Multiple Sensors on Different Body Positions. In *Proceedings of the Int'l Workshop on Wearable and Implantable Body Sensor Networks, BSN'06*, pages 113–116. IEEE CS, 2006.
- [9] C. Park, P. H. Chou, and Y. Sun. A wearable wireless sensor platform for interactive dance performances. In *Proceedings of the 4th IEEE Int'l Conference on Pervasive Computing and Communications, PerCom 2006*, pages 52–59. IEEE CS, Mar. 2006.
- [10] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. Activity recognition from accelerometer data. In *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence, IAAI'05*, pages 1541–1546. AAAI Press, July 2005.
- [11] D. Roggen, A. Calatroni, M. Rossi, T. Holleczeck, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, J. Doppler, C. Holzmann, M. Kurz, G. Holl, R. Chavarriaga, M. Creatura, and J. del R. Millán. Collecting complex activity data sets in highly rich networked sensor environments. In *Proceedings of the 7th International Conference on Networked Sensing Systems, INSS 2010*, Kassel, Germany, June 2010. IEEE CS.
- [12] D. Roggen, K. Förster, A. Calatroni, T. Holleczeck, Y. Fang, G. Tröster, P. Lukowicz, G. Pirkl, D. Bannach, K. Kunze, A. Ferscha, C. Holzmann, A. Riener, R. Chavarriaga, and J. del R. Millán. OPPORTUNITY: Towards opportunistic activity and context recognition systems. In *Proceedings of the 3rd IEEE WoWMoM Workshop on Autonomic and Opportunistic Communications, AOC 2009*, Kos, Greece, June 2009. IEEE CS.
- [13] P. van de Ven, A. Bourke, J. Nelson, and G. O. Laighin. A wireless platform for fall and mobility monitoring in health care. In *Proceedings of the 3rd Int'l Conference on Body Area Networks, BodyNets'08*, pages 1–4. ICST, Mar. 2008.
- [14] F. Zhu, M. W. Mutka, and L. M. Ni. Service discovery in pervasive computing environments. *IEEE Pervasive Computing*, 4:81–90, 2005.