The OPPORTUNITY Framework and Data Processing Ecosystem for Opportunistic Activity and Context Recognition

Marc Kurz[*,1], Gerold Hölzl[1], Alois Ferscha[1], Alberto Calatroni[2], Daniel Roggen[2], Gerhard Tröster[2], Hesam Sagha[3], Ricardo Chavarriaga[3], José del R. Millán[3], David Bannach[4], Kai Kunze[4] and Paul Lukowicz[4]

[1]*Institute for Pervasive Computing, University of Linz, Austria*

[2]*Wearable Computing Laboratory, ETH Zurich, Switzerland*

[3]*EPFL, STI-CPN-CNBI, Lausanne, Switzerland*

[4]*Embedded System Lab, University of Passau, Germany*

Abstract: Opportunistic sensing can be used to obtain data from sensors that just happen to be present in the user's surroundings. By harnessing these *opportunistic sensor configurations* to infer activity or context, ambient intelligence environments become more robust, have improved user comfort thanks to reduced requirements on body-worn sensor deployment and they are not limited to a predefined and restricted location, defined by sensors specifically deployed for an application.

We present the OPPORTUNITY Framework and Data Processing Ecosystem to recognize human activities or contexts in such opportunistic sensor configurations. It addresses the challenge of inferring human activities with limited guarantees about placement, nature and run-time availability of sensors. We realize this by a combination of: (i) a sensing/context framework capable of coordinating sensor recruitment according to a high level recognition goal, (ii) the corresponding dynamic instantiation of data processing elements to infer activities, (iii) a tight interaction between the last two elements in an "ecosystem" allowing to autonomously discover novel knowledge about sensor characteristics that is reusable in subsequent recognition queries. This allows the system to operate in open-ended environments.

We demonstrate OPPORTUNITY on a large-scale dataset collected to exhibit the sensor richness and related characteristics, typical of opportunistic sensing systems. The dataset comprises 25 hours of activities of daily living, collected from 12 subjects. It contains data of 72 sensors covering 10 modalities and 15 networked sensor systems deployed in objects, on the body and in the environment. We show the mapping from a recognition goal to an instantiation of the recognition system. We also show the knowledge acquisition and reuse of the autonomously discovered semantic meaning of a new unknown sensor, the autonomous update of the trust indicator of a sensor due to unforeseen deteriorations, and the autonomous discovery of the on-body sensor placement.

Keywords: Opportunistic sensing, activity and context recognition, framework and data processing.

1. INTRODUCTION AND MOTIVATION

1.1. Background

Sensor networks (integrated into objects, on the body or in the environment) allow to sense the physical world and persons acting in it [1-3] which is a key to devise intelligent environments [4]. Human activity is an important aspect of context [5]. Activity recognition is used in pervasive computing [6], wearable computing [7] and in human computer interaction (HCI) [8]. It opens the way to systems capable of pro-actively supporting users with just-in-time assistance, systems responding to natural interactions, or systems mining daily life patterns. A few applications include gestural mobile interfaces [9], Industrial workers assistance [10, 11], translation of sign language [12], rehabilitation [13], healthcare [14], support for elderly people [15], or wellness [16]. Activities span low-level actions such as modes of locomotion (walking, running, standing), postures (sitting, lying), gestures (reaching an object, opening a door), and higher-level composite activities made of sequences of actions (e.g. preparing a breakfast consists of a statistically characteristic sequence of actions). Our focus lies on these low-level actions, as they are the foundation to recognize higher-level composite activities.

A wide range of sensors can be used for activity recognition, including sensors placed on the body, in objects, or in the environment. Commonly used sensors include body-worn movement sensors (accelerometers, inertial measurement units), presence sensors, localization systems, reed switches, RFID tags and sound sensors. We refer to [17] for a summary of other sensors used in activity recognition.

In wearable and pervasive computing, activity recognition is tackled as a problem of *learning by demonstration* [18, 19]. Meaning is attributed to the sensor data streams by "comparing" them to "known signals". This mapping from

*Address correspondence to this author at the Institute for Pervasive Computing, University of Linz, Austria; Tel: +43-732-2468-8527; Fax: +43-732-2468-8426; E-mail: kurz@pervasive.jku.at

2210-3287/11 $58.00+.00

sensor signals to activity or context classes is done using machine learning techniques or information retrieval techniques. The "known signals" are defined at design-time based on a training dataset comprising the sensor signals corresponding to activities or contexts elicited in a training environment.

Consequently, there are strict requirements imposed on the designer of ambient intelligence environments, and the users of such systems. Sensors are deployed at design time for specific applications. At run-time, the system requires exactly the same set of sensors as initially foreseen. This forces users of a wearable assistant to place sensors day after day on the same body locations, which is cumbersome. Sensorized garments must be tight fitting to minimize variability, making them impractical for many applications, such as rehabilitation or assistance of elderly people. Sensors must be identically deployed in different smart buildings. Overall, these constraints limit the widespread diffusion of activity and context aware systems. Any change in the sensor characteristics or placement leads to a change in the sensor signal to activity/context class mapping. The result is degraded recognition performance or complete failure. Such changes occur, for instance, when sensors are displaced or rotated on the body (e.g. acceleration sensors in a mobile phone, when the phone can move in a pocket), or when signal quality degrades (e.g. due to wireless occlusions or running out of energy).

1.2. Opportunistic Activity and Context Recognition

For a widespread use of activity-aware and context-aware systems, application specific sensor deployment is not desirable. Sensing is better seen as *opportunistic* [20]: the sensors available at any point in time should be best exploited for a recognition task. We refer to this as activity recognition in *opportunistic sensor configurations* or *opportunistic activity recognition*. We envision opportunistic activity recognition based on the following grounds [21]:

- ***Increasing Availability of Resources***

Due to technological advancements, sensor systems are becoming smaller and smaller and due to their vast heterogeneity and to the (wireless) communication capabilities, devices that measure different environmental quantities can be embedded and integrated in different kinds of electronic appliances and gadgets [2]. Thus, future environments will see an ever-larger availability of readily deployed sensors. The newest generation of smart-phones, for example, can be seen as a multi-sensor platform, as most of them are equipped with position sensors (GPS), acceleration, orientation, light, noise and temperature detection. Digital cameras, with their capabilities of automatically tagging photographs with their global position, can be easily utilized as positioning sensors. Another interesting item that shows the trend of integrating sensor technology into conventional objects is the *Texas Instruments eZ430 Chronos*[1]. Besides the usual functionality, this watch is capable of sensing the acceleration and the surrounding temperature. Sensors are also available in toys, in building automation systems (e.g. to detect door/windows being opened or closed), in furniture, in recently invented "robotic objects" and even in some garments [22] and sports shoes.

A number of these sensors are specifically dedicated to activity recognition, usually in special assisted living houses. However, the vast majority of existing sensors have been foreseen for other uses. Yet, these sensors can often be re-purposed for activity recognition. For instance, proximity infrared sensors are typically used to turn on lighting automatically, but they can be re-purposed to distinguish static postures from user movement. A discussion on sensor re-purposing is found in [23].

- ***Open-Ended Environments***

While some sensors may be known to be available (e.g. integrated in all clothings in the same location and with the same characteristics across all brands), it is much more likely that in a real-world deployment of activity-aware systems the nature, type, availability of sensors will be highly dynamic and hard to predict. This will depend on the clothes that the user wears (different clothes may offer different sensors), the sensorized gadgets that the user takes with him or leaves behind (e.g. mobile phone, hearing instrument, PDA, sensorized watch), and his location and surroundings. Typically, different rooms will offer different sensing capabilities. For instance, a conference room may be equipped with cameras for video-conferencing, manufacturing environments may be equipped with presence sensors to shut down machinery in case of danger, while a bed may measure the user's heart rate during sleep. Such environments are open-ended as they change over time through upgrades in unpredictable ways.

Opportunistic sensing has the potential to deliver much more data than the current statically deployed sensor setup, by networking all available resources surrounding the user. The potential benefits include: freeing activity-and context-aware applications from operating in conscribed and specifically designed ambient intelligence environments, improving activity recognition performance, increasing the robustness of ambient intelligence environments, and improving user comfort as a consequence of more flexible requirements on the deployment of body-worn sensors.

However, opportunistic sensing poses a new class of challenges for activity recognition: *sensor data must be interpreted without assuming a-priori known sensor sets, and to a larger extent without assuming that sensor signal to context mapping is known at design time.*

1.3. Contribution

We present a framework and data processing eco-system called OPPORTUNITY devised for activity recognition in opportunistic sensor configurations (see Fig. **1**).

The OPPORTUNITY framework plays the role of a sensing and context framework coordinating sensor recruitment according to a high-level recognition goal using a combination of activity modeling and sensor self-description capabilities. It runs on a user's mobile device and on the sensor nodes. It is capable of updating its knowledge at runtime about the sensor capabilities thanks to a tight interplay with the data processing methods.

[1]http://processors.wiki.ti.com/index.php/EZ430-Chronos

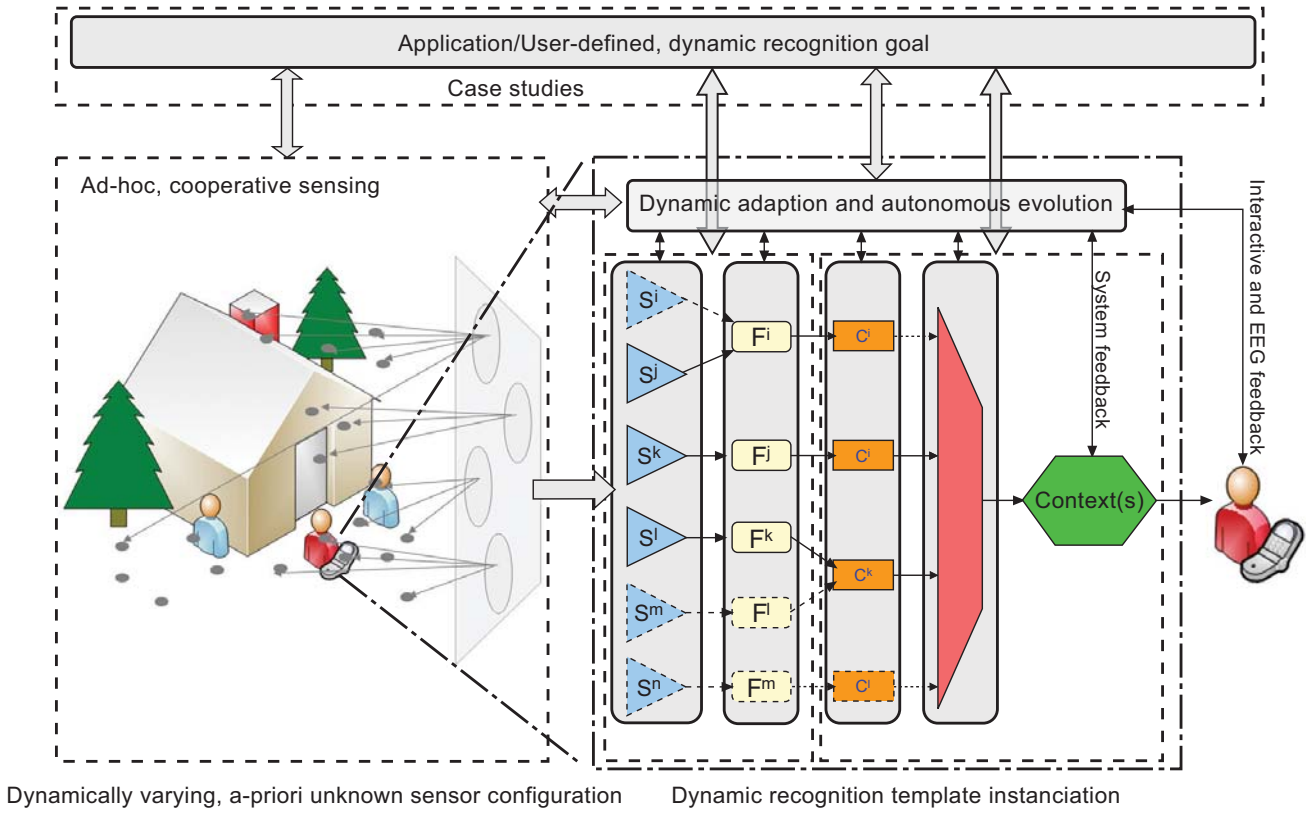

Fig. (1). The OPPORTUNITY Framework and Data Processing Ecosystem: the user's mobile device triggers the sensor nodes self-organization. Each sensor node (a Context Cell) is an autonomous unit capable of self- description and other self-* properties that infers the user's context from the sensor data. It can update its probabilistic context representation (online learning) from neighbors' inputs, share it with the mobile device, and update its self-description, thus forming an autonomously evolving and adapting sensor ecology. High-level goal description and formulation language play together with sensors self-* capabilities to trigger a self-organization of the sensing network for the selected recognition goal.

The data processing methods use the information provided by the framework to dynamically instanciate the appropriate processing elements (feature extraction, classifiers, data fusion) to infer activities. They are designed to tolerate the large sensor-signal to activity-class variability resulting from opportunistic sensing.

A key aspect of OPPORTUNITY compared to other approaches is that the framework and data processing methods are designed to complement each other in a so-called "ecosystem". Initially, the framework contains the design-time-defined knowledge and allows to perform an initial selection of the sensors relevant for a recognition task. The recognition itself is done with signal processing and machine learning techniques. Furthermore, the data processing techniques allow to aquire new knowledge at run-time and inform the framework about it. The framework can then exploit this in subsequent activity recognition queries.

The new knowledge that can be acquired includes currently the autonomous detection of sensor data anomalies, the autonomous discovery of the semantic meaning of unknown new sensors, and the autonomous discovery of on-body sensor placement. Further knowledge discovery are envisioned in [21] but not covered here. Anomaly detection such as sensor signal degradation (e.g. caused by a user displacing a sensors on-body) is conveyed to the framework, which reconfigures the sensing ensemble to fulfill the recognition goals. Autonomous learning of the semantics of new sensors occurs by translating the recognition capabilities of an existing sensor onto a new sensor and quantifying the resulting performance through a process of transfer learning, or by quantifying statistical signal similarity. Autonomous on-body sensor placement discovery is realized by pattern classification on the sensor node data. The new knowledge that is gained is stored and advertised through the sensor self-description. This allows the framework to expand or refine its knowledge about the capability of the surrounding nodes at run-time. Thus, the framework may refine or reshape the sets of sensors used to fulfill a recognition goal.

We present the opportunistic system design tool-chain used to define the static sensor characteristics from training datasets and allowing to train initial activity recognition models when design-time knowledge is available. This allows to provide the initial knowledge to the system.

We demonstrate OPPORTUNITY on a large-scale dataset collected to exhibit the sensor-rich characteristics of opportunistic sensing systems. The dataset comprises 25 hours of activities of daily living, collected from 12 subjects. It contains the data of 72 sensors of 10 modalities and part 15 networked sensor systems deployed in objects, on body and in the environment. We show the mapping from a

recognition goal to an instantiation of the recognition system, and the new knowledge acquisition and reuse.

1.4. Outline

We present related work in sensing and context frameworks and data processing for activity recognition in Section 2. In Section 3 we describe the emergent sensor settings in everyday artifacts, where we introduce the concepts of sensor self-descriptions and sensor abstractions for commonly accessing different sources of environmental information. In Section 4 we present the framework part of the OPPORTUNITY ecosystem, which is a prototypical implementation of a mobile opportunistic activity and context recognition system. There we show the conceptual architecture that builds the base for the implemented prototypical solution, we explain how sensors can describe themselves with respect to their sensing capabilities according to recognition goals, and we provide a description of how a recognition goal is processed in the system. Section 5 highlights some specialized machine learning technologies that target the opportunistic requirements (run-time discovery of new knowledge). In Section 6 we present support tools used to define sensor properties (e.g. self-description) from large training datasets. We present how the data processing part of the OPPORTUNITY ecosystem interfaces with and integrates within the framework in a task of recognition of modes of locomotion in Section 7. We show features that are especially important for opportunistic activity and context recognition system, namely the handling of, (i) sensor appearing, (ii) sensor disappearing, (iii) transfer learning of recognition capabilities during runtime, and (iv) detecting anomalies in the sensor data. We discuss the implementation and conclude this paper with an outlook in Section 8.

2. RELATED WORK

At a networking level, opportunistic sensing is generally understood as carrying data in the absence of static infrastructure and simultaneous end-to-end communication paths. Instead, mobile nodes or mules carry data from cluster to cluster [24-27]. Sensor data are collected on the way and disseminated in an opportunistic fashion [28]. We refer to [20] for recent developments in opportunistic sensing.

Here we understand opportunistic sensing as the process of acquiring and interpreting data from sensors that *just happen to be available in the surroundings of the user* to eventually infer human activities and context. Thus, our work relates closely to *sensing and context frameworks*, and *data processing techniques for activity recognition*. Low-level network self-organization (e.g. node discovery, MAC protocols, decentralized routing) is also required, but it is not the focus of this paper. We assume that this is readily available. Protocols such as Zigbee offer basic functionalities that support the realization of the system presented here. A number of more advanced approaches exist for the coordinated emergence of sensing networks [29, 30].

2.1. Sensing and Context Frameworks

Sensing and context frameworks attempt to streamline, abstract and mainstream the management, processing, and reasoning on a large number of heterogeneous and dynamic resources. Pervasive connectivity leads towards an "Internet of Things" or a "Real-World Internet", where any physical resource (sensor or actuator) can be interfaced from the Internet. Sensing and context frameworks have been proposed to harness such massive amount of resources [31]. However, the focus lies mostly on infrastructure and interoperability aspects.

Recently, frameworks were proposed to support urban sensing, participatory sensing and crowd sourcing [32-36]. They have been devised for the recognition of *collective* human behaviors. This includes e.g. the detection of mass transit phenomena in a city, the analysis of patterns of people commuting, or the retrieval of places of interest, based on people movement. This form of collective activity recognition is not suitable for our needs. Our focus lies on the activities of single individuals, such as modes of locomotion (e.g. walking, running), posture (lying, sitting, standing, etc.), and fine-grained gestures (e.g. reaching an object, drinking from a cup etc.).

On a smaller scale, "personal" sensing frameworks have been proposed to manage and acquire data through a mobile phone, which is the core component of those systems. These frameworks are usually more suitable for fine-grained activity recognition, as they are dedicated to managing the resources in the user's personal area network. There are frameworks devised specifically for data acquisition [37] or for the acquisition of sensor data on the user's body and their interpretation in terms of human activities [38, 39]. These systems assume that the set of sensors defined at design time is continuously available also at runtime, in order to operate properly.

Most efforts towards handling a dynamic availability of resources consist in performing a run-time exchange of sensors or in remapping a processing tree according to available resources. This approach is followed by the Titan framework, which is a service-oriented architecture able to deploy activity or context aware applications in a dynamic and heterogeneous personal area network [40]. It can map at runtime applications made of services composed on the dynamically discovered resources. However, it can only operate if the exact set of used services is available in the user's surroundings. Thus it cannot capitalize on opportunistic sensing, unless a vast amount of expert-knowledge is included in the system at design time and activity recognition services are provided for each possible available sensor.

Numerous other toolkits, middlewares and frameworks were proposed to design context-aware applications [41-44]. For a recent review we refer to [45, 46]. Many of these systems make use of ontologies to model context and rely on a reasoner to infer the user's context from semantically meaningful events provided as inputs. For activity recognition, events may be e.g. the mode of locomotion of the user. Ontology-based frameworks abstract the means by which the events are obtained. This allows to make reasoning at a higher level, to be independent of the exact source of the events. However, these systems do not address the interpretation of sensor data that leads to the generation of these events. This is one key challenge when dealing with activity recognition with opportunistic sensor configurations.

2.2. Data Processing for Activity Recognition

Context toolkits assume semantically meaningful events as input (e.g. modes of locomotion). Inferring these events from sensor data streams requires signal processing and machine learning techniques applied on raw sensor data. This has been investigated in wearable and pervasive computing as a problem of *learning by demonstration*. The *activity recognition chain* (ARC) is a set of processing principles commonly followed by researchers to infer human activities from the raw sensor data [18, 19, 47, 48]. Meaning is attributed to the sensor data streams by "comparing" them to known activity prototypes. This is realized by signal processing and machine learning techniques. The processing stages are usually: (i) *Sensor-data acquisition* - a stream of sensor samples is obtained; (ii) *Signal pre-processing* - the sensor data stream is pre-processed, which typically involves transformations like calibration, de-noising or sensor level data fusion; (iii) *Segmentation of the data stream* - the data stream is segmented into sections that are likely to contain a relevant activity; (iv) *Feature extraction* - features are computed on the identified segments to reduce their dimensionality, yielding a feature vector which is usually much smaller than the number of samples; (v) *Classification* - a classifier maps the feature vector into a pre-defined set of output classes, i.e. activities, gestures or context; (vi) *Decision fusion and higher-level reasoning* - multiple sources of information, like multiple sensors, or multiple classifiers operating on one sensor, are combined to produce a decision about the activity that occurred; (vii) *"Null-class" rejection* - in cases where the confidence in the classification result is too low, the system may discard the classified activity.

Various methods can be used at each stage. However, the condition which is always taken for granted is that the mapping between sensor signals and activity classes remains identical at runtime compared to design time. Thus, at design time, a set of sensors is foreseen and deployed and it is assumed that these sensors remain available during runtime. Some tolerance to variability is achieved by acquiring training data from a large number of people, who are inherently performing the activities in a slightly different manner. Some approaches were proposed to tolerate variability in sensor placement [49-51]. However, these approaches tolerate small variations compared to the ones likely occurring in realistic scenarios. The variations, which are typically allowed, are limited to displacement on a limb for fine-grained activities.

2.3. Summary

None of the related work addresses the issue of attributing meaning to sensor readings, in the absence of design-time knowledge about the sensor availability and characteristics.

Current sensing and context frameworks provide reconfiguration capabilities in case of change in the sensing network. However, they do not consider the problem of the interpretation of the delivered data, after a reconfiguration. In an opportunistic setup, such reconfigurations are usually unforeseen at design time. Current data processing techniques aim at increasing the tolerance to sensor variability for activity recognition. However, they focus on the "small" variations, which typically occur in statically deployed sensor configurations, such as the slip of a sensor on a limb, or the change in the orientation of a sensor integrated in clothing. They fail to realize that opportunistic sensing has the potential to deliver much more data than a statically deployed sensor setup, and thus they do not address the type of variability in sensor data to activity class mapping that results from opportunistic sensing.

In this work we argue for a greater integration of the sensing/context frameworks together with the data processing into an "ecosystem" with each element designed in order to support activity recognition in opportunistic sensor configurations. Thus, the needs of the data processing algorithms can be reflected into the sensing/context framework (e.g. by providing sensor self-descriptions) and the needs of the framework can be supported by data processing algorithms (e.g. by learning at runtime the semantic meaning of data delivered by newly discovered, a-priori unknown sensors).

3. REVERSING THE CONTEXT ARCHITECTURE

3.1. Emergent Sensor Settings

There is an emerging plethora of sensor-rich devices in the environment. This makes the explicit deployment of sensing devices for activity and context recognition dispensable. The challenge shifts from the deployment of sensing devices and the design of the associated activity and context recognition systems (e.g. recognition chains), to the identification, access and utilization of the sensing devices that already exist in the environment. In order to use unknown sensor resources, their capabilities and characteristics have to be known. Therefore, the sensor characteristics, purposes and capabilities have to be described and this information needs to accompany the sensor systems. This is implemented in the OPPORTUNITY approach with the help of sensor self-descriptions. These are based on the standardized markup language SensorML [52], which allows to describe sensing devices. Those XML documents enable the semantic matching and querying of sensors for specific activity recognition goals. How the sensor self-description looks like in detail in the OPPORTUNITY Framework and how it is applied to enable goal-driven opportunistic sensing is explained in detail in Sections 3.3 and 4.3. Furthermore, as different sensing devices are different in the way they are accessed and handled, an opportunistic system has to define a common way to access heterogeneous data-delivering entities. Therefore, we introduce the concept of sensor abstractions in the following Section 3.2.

3.2. Federating Available Sensor for a Purpose

As discussed in the previous section, the count of available and accessible devices that deliver environmental measurements in everyday artifacts is constantly increasing. Furthermore, sources of information exist that are not of physical nature, but can be logical or virtual devices [53]. Different sensor systems (of different types) have different working characteristics, they measure different environmental quantities, deliver different types of data and they might be accessible and controllable by a system or an application in different ways. Therefore, to have a simple

and common standardized access to different sensor devices, wrappers are needed, which encapsulate hardware details and hide the low-level access details (e.g. direct memory access, data transmission, etc.) that might be dependent on the kind of device. Thus, a wrapper is a software abstraction of a sensor device that provides the complete functionality to a system by hiding the complexity. In [54] the authors present the *Context Toolkit*, where the concept of context widgets is introduced, which is very similar to the aforementioned wrappers. A context widget is defined as a component that provides applications with access to contextual information from their operating environment. The context widgets hide the complexity of the sensor systems, they abstract context information and they provide reusable building blocks for a system. All those characteristics can also be taken as mandatory for the sensor abstractions (the wrappers) in an opportunistic activity and context recognition system. Additionally, we further abstract the sensor systems in different types as given by their working characteristics and their practicability and suitability, because dealing only with a set of physical sensor abstractions is not satisfactory in an opportunistic activity and context recognition system. The following list provides a description of some of the different sensor types that have been identified [55]:

- ***PhysicalSensor***

This type is the common physical device, which is mostly composed of (i) a processing unit, (ii) memory, (iii) a power supply, and (iv) an interface for enabling (wireless) communication. The challenge for the sensor abstraction is to hide the low-level hardware details, to have all physical sensor devices available and accessible in an opportunistic activity and context recognition system via a common interface. Examples for *PhysicalSensors* are the aforementioned *Texas Instruments eZ430 Chronos*[2], a SunSPOT (Small Programmable Object Technology[3], or the InterSense InertiaCube3[4].

- ***OnlineSensor***

Environmental data are not necessarily delivered by a physical device, an online accessible source of information (e.g. a webservice) might also be considered valuable in an activity and context recognition system. This type does not rely on a physical device connected to a computer. An example could be a webservice on the Internet that provides regional weather information (e.g. *Yahoo! Weather*[5]). Instead of utilizing a physical device that measures the humidity and temperature, this online accessible source of information acts as a sensor as it is abstracted as type *OnlineSensor*. Therefore, the abstraction deals with the connection to the remote source, the data acquisition, transmission, and processing.

- ***HarvestSensor***

We define a *HarvestSensor* as a physical device that autonomously collects environmental data, stores these data locally on some internal memory and provides these recorded data when the system puts the sensor in replay mode. An example for such a device is the *ActiGraph GT3X*[6] sensor. It provides activity measures, like steps taken and energy expenditure from persons, and is equipped with a 4MB flash memory, that is capable of storing data for more than one year without having the system attached to a power supply.

The concept of sensor abstractions is depicted in Fig. (**2**). The different symbols illustrate the different types of sensors (e.g. *PhysicalSensor*, *PlaybackSensor*, and *OnlineSensor*). Each abstract sensor type will be accompanied by its self-description (see also Fig. **6**), which is an inevitable item in an opportunistic system (for further details on the sensor self-descriptions and their application in an opportunistic system, refer to Section 3.3 and to [56, 57] and [55]). The surrounding lucent light-blue circle indicates the abstraction of the sensing device, which enables the common usage regardless of which type the sensor system is. From the opportunistic system's point of view, every environmental data-delivering entity is a *Sensor* as shown in Fig. (**2**).

3.3. Goal Language and Self-Describing Sensors

A recognition goal is a high level principle governing how a system should behave [58]. Goals state in an abstract way what the system should do, how it should behave and how this can be achieved. Thus, goals are a way of controlling the behavior and the configuration of a system. Using high level goals is a novel approach to direct the configuration of a system in the field of sensor networks. As the available sensor infrastructure and its configuration is not known at design time and due to its highly dynamic nature during runtime (sensors can appear or disappear) the goal has to define what is necessary to achieve its purpose.

In order to cope with an infrastructure, which is changing over time, it is important to avoid matching the sensor capabilities in a static way to the stated recognition goal, as this would restrain the dynamic nature of the opportunistic sensing approach. Therefore, there is the need of defining a syntax and a semantic mapping between a high-level recognition goal and the sensors, to be able to identify sensor candidates that can contribute to the stated high-level goal. Two major aspects have to be taken into account when devising this mapping: (i) we need a *semantic description* of the sensors and their capabilities, stating for which goals and to which extent the sensors can be used (the sensors self-description) and (ii) a knowledge base that models for a given application domain the available goals and their relationships. A goal that is defined in the knowledge base can be stated to the system as a recognition goal. If there are sensors available in the sensing infrastructure that can contribute to the given goal, these sensors are selected and combined into an ensemble to contribute to the stated recognition goal. If no sensors are available, the stated goal is refined into its sub goals/activities according to the relationships defined in the knowledge base. After the refinement process, the system has to search if there are sensors available for the refined sub goals. This can of course involve the recursive evaluation of an entire tree of goals and sub goals and can either result in an ensemble of sensors that can fulfill the

[2] http://processors.wiki.ti.com/index.php/EZ430-Chronos
[3] http://sunspotworld.com/
[4] http://www.intersense.com/pages/18/59/
[5] http://weather.yahoo.com/
[6] http://www.theactigraph.com/

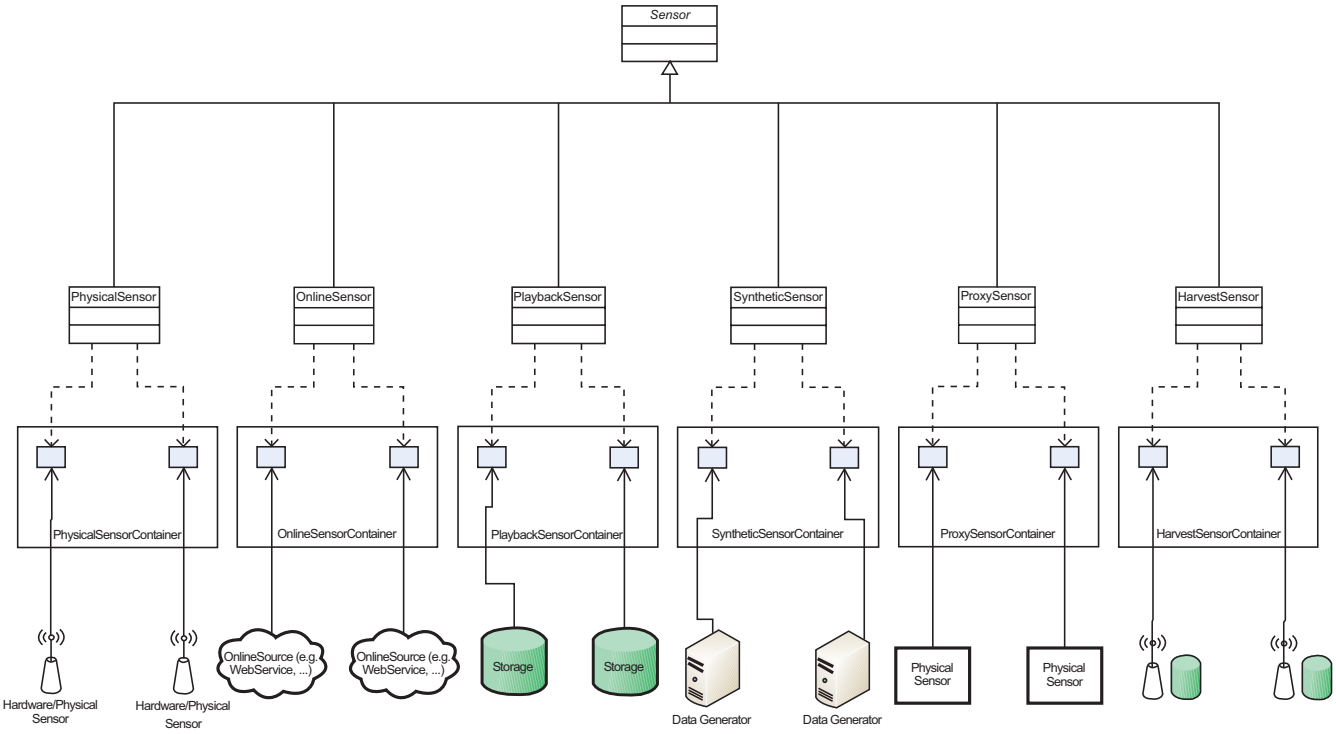

Fig. (2). The concept of sensor abstractions: every data-delivering entity in the environment can be accessed in a common way as *Sensor*. The abstracted types that are shown in this Figure are (**i**) *PhysicalSensor*, (**ii**) *OnlineSensor*, (**iii**) *PlaybackSensor*, (**iv**) *SyntheticSensor*, (**v**) *ProxySensor*, and (**vi**) *HarvestSensor*. From the system's point of view a data delivering source is accessed as Sensor with one common interface [55].

goals (and/or the refined sub goals) or it can produce no results.

In the OPPORTUNITY approach, the self-descriptions of the abstracted sensors are segmented into two parts (both following the standardized SensorML [52] markup language): (i) the static technical description (contains physical properties, description of the communication interface, etc.), and (ii) a dynamic part which is used to define the sensor capabilities according to a recognition goal [56, 57]. This dynamic part can be utilized to query the available sensors for a stated recognition goal to find the set of candidates that are able to contribute. Fig. (**3**) shows the concept of the segmented self-description.

The technical part of the description contains static entries that are equal for all sensors of the same type. Examples for these technical entries are the sensor's physical properties (width, length, height), its working characteristics, its communication interface, and power characteristics (the technical description of a sensor can be seen as a SensorML translation of the datasheet). The dynamic self-description of the sensor capabilities according to a recognition goal in a certain domain (shown on the left-hand side of Fig. (**3**)) contains, (i) the unique identification of the sensor system, (ii) the sensor capabilities (defines to which goal it can contribute to and to which extent), (iii) the configuration (how has the sensor to be configured to be capable of contributing to a recognition goal), (iv) the plausibility/trustworthiness of the delivered sensor data, and (v) historic contributions to recognition goals. The dynamic part of the sensor self-descriptions is stored in the so-called *ExperienceItems*. Section 4.3 provides a detailed description of how these items are structured, what elements are contained in detail and how they can be applied within the opportunsitic activity and context recognition system.

For describing goals, we use a similar approach as presented by *Ayomi Bandera et al.* [59] where the authors propose to use the *Web Ontology Language (OWL)*[7] to have an effective semantic matching to describe the requests (in our case the goals) and the advertisements (the abilities of a sensor) to define a goal language. A request typically consists of several individual requirements to be satisfied. Each requirement is made of a description stating which resource characteristics are needed (e.g. the fulfillment of a certain goal), and the priority or weight of the requirement itself. The resource providers (sensors) will specify all the relevant characteristics, advertising them by means of the self-descriptions. In our approach, the ontology only holds the goals and their semantic relationships and no further information about the sensors that can be utilized. The goals a sensor can be used for are stored by the sensor itself in its

[7] http://www.w3.org/TR/owl-features/

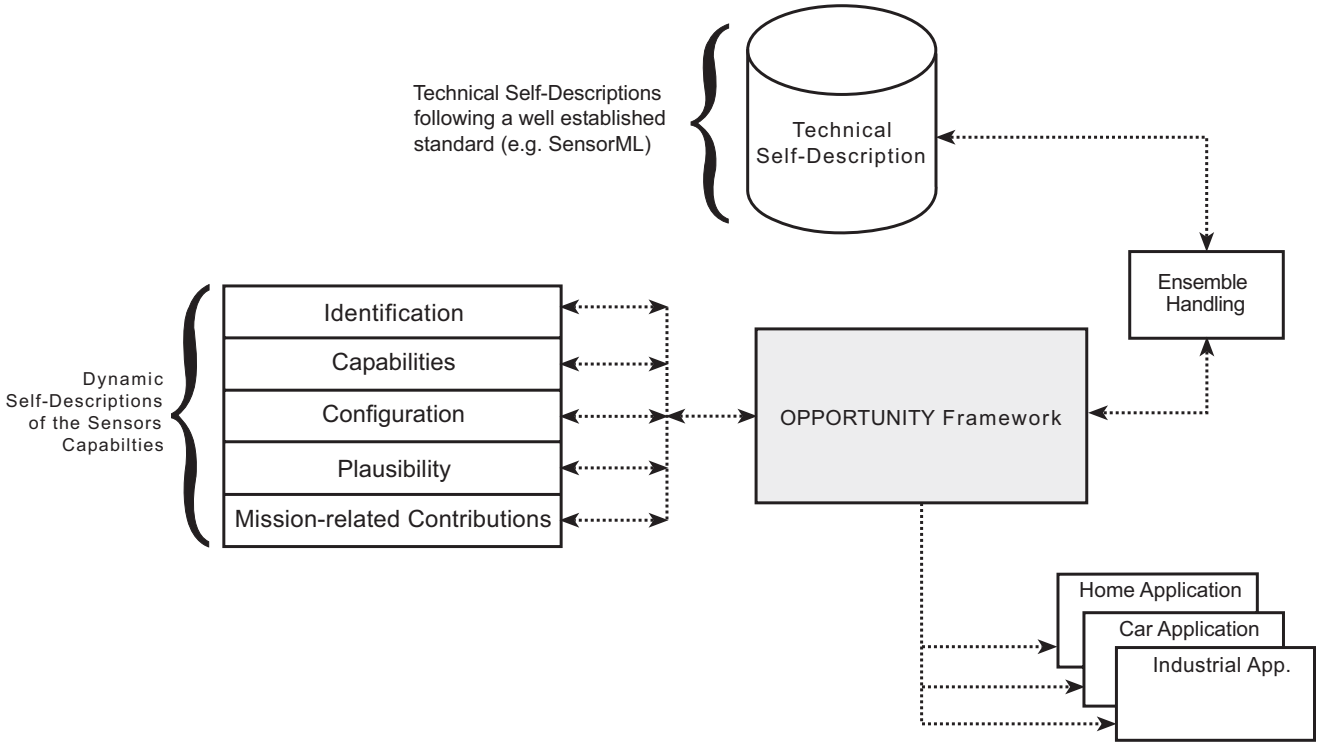

Fig. (3). The segmented self-description concept [56]. The technical (static) part of the self-description is depicted at the very top. The dynamic part (left-hand side) contains different changeable items that are modified during the sensor's lifetime and it stores the active contribution to a recognition goal, including, (**i**) capabilities with respect to the recognition goal, (**ii**) necessary sensor configuration and (**iii**) plausibility of the delivered sensor datastream (keeping track of changes due to sensor rotations, displacement, or other malfunctions).

self-description. A request to the framework to detect a goal may either consist of one single goal (e.g. `WALK`) or of several individual goals (e.g. `SIT`, `STAND`). Each stated goal has to be an entity in the used ontology. This common knowledge of the goals and their relations ensures the correct mapping of the sensors, as the sensors know from their self-description to which goal they can contribute.

The priority (weight) of the single requirements is not specified using a number, we instead introduce a *usingType* with the values *uses* and *canuse*. These two values indicate if a goal in a request must be satisfied (*uses*) or has to be only optionally satisfied (*canuse*). As an example, if in our model making breakfast involves the user drinking and eating, sitting at a table and sipping from a cup, then to detect `BREAKFAST` the sub-goals `DRINK`, `EAT`, `SIT` and `SIP CUP` must be satisfied, so they have to be linked with the (*uses*) directive. If the goal Locomotion should be detected, it can be satisfied by detecting only one, some or all of its related activities (e.g. `LIE`, `RUN`, `SIT`, `WALK`). Therefore, these will be linked to the main goal with the (*canuse*) directive. In summary, the *usingType* indicates if the considered goal or sub-goal is a mandatory one, i.e. if the goal should be strictly satisfied in the capabilities of a sensor, in order for the system to consider that sensor as a potential match. The goals that are defined in one request are combined using a logical *AND* connection. In order to sense for different, independent goals, there is the need of a logical *OR* connection (e.g. detect `WALK` or `SIT`) [60]. The processing of the request is done by recursively combining the goal-requests (also referred to as *SensingMission*) split by a " + ". This enables the sensing for independent goals e.g detect (`WALK + SIT + LIE`).

In general, a goal request can take the form:

$$SensingMission = Goal\ \{Goal\}\ [+SensingMission];$$
$$Goal = NamedGoal["["usingType"]"];$$
$$NamedGoal = "GoalFromOntology";$$
$$usingType = uses" \mid "canuse";$$

As the values of *NamedGoal* are defined in the used knowledge base, the EBNF representation of a goal only states a symbolic link (*GoalFromOntology*) to the goals that are defined in the ontology. This goal request approach allows the user to state one or more recognition goals that the system should detect. Having the possibility of defining such conditions in a programming-like style from the user's side gives the maximum flexibility in defining what the system should recognize.

Besides the syntax of how a recognition goal can be formalized and stated to the system, there is the need to represent a goal internally in the system for processing it. The internal representation is based on an XML-data structure that holds the information for a given sensing mission. The structure encapsulates which goals need to be detected to satisfy the requested goal and if they are mandatory or not. This description is built upon the standardized SensorML [52] to be compatible with the sensor self-descriptions. The stored information contains in detail:

- The *ID* of the SensingMission. As there can be multiple sensing missions running at the same time, there is the need to distinguish between the goal representations of each sensing mission.
- A *GoalList* that holds the information about which goals have to be detected to fulfill the mission and if they are mandatory or not.

As the refinement process can split up the given goal into sub-goals and satisfy them instead of the stated goal, the goal list can have as many entries as the refinement process states to be necessary to achieve the sensing mission. If for example the goal `LOCOMOTION` is stated to the system but no sensor is capable of contributing to this goal, it is refined into (`WALK + SIT + LIE + STAND + RUN`) according to the knowledge base (ontology). The goal description still holds the same ID (*LOCOMOTION*) as before the refinement process took place, so it belongs to the same sensing mission, but the goals to be satisfied in the *goalList* and their *usingType* have been altered (`LOCOMOTION -> LIE, RUN, SIT, STAND, WALK`) according to the semantic relations of the goals stored in the ontology.

4. THE OPPORTUNITY FRAMEWORK

4.1. Overview

This section describes the OPPORTUNITY Framework, which is a prototypical system that integrates technologies that are applied for opportunistic activity and context recognition together with methodologies to describe, translate and process recognition goals that are stated by users and/or applications to the system at runtime. The framework acts as fundamental basis for the design, analysis and development of systems able to recognize complex activities and contexts in an opportunistic way. Furthermore, it also represents the testbed and development environment for fully functional cooperative, goal-oriented and opportunistic sensing applications. Firstly, a conceptual architecture for opportunistic activity and context recognition is described, which builds the base for the implemented framework. Then, the concept of self-describing sensors is further described, together with the methods to enable sensor querying according to a recognition goal. As a recognition goal can be stated to the system at runtime, the goal processing is also a relevant part of the framework, together with the semantic modeling of recognition goals and activities in form of an ontology.

4.2. Architectural Concept

The basic understanding of a "software framework" is a run-time environment together with a code base and libraries, able to run autonomously on a target platform. In the OPPORTUNITY case, the run-time environment aims at being portable to a variety of small and tiny hardware platforms, thus having small memory footprints. The runtimes of various different network nodes must be able to communicate, share, cooperate, and coordinate. The OPPORTUNITY Framework is implemented in Java[8] and OSGi[9]. We use the OSGi framework to build the OPPORTUNITY Framework for several reasons, namely (i) the universality of code deployment, (ii) the life cycle management abilities of OSGi, (iii) the modularity, component oriented and object-oriented paradigm and (iv) the portability and thus compatibility with various different hardware platforms.

The implementation builds upon the architectural concept, which is depicted in Fig. (**4**), which shows the concept and general activity and context recognition chain for an opportunistic architecture. In this illustration, two different goals ("I need"-Requests) are passed on to the system. At the top, the users and applications can formulate a recognition goal in an abstract manner that is handed to the system. This request is translated into a machine-readable expression (the sensing mission, as described in Sections 3.3 and 4.4). According to this mission, the available self-describing sensors (at the very bottom of Fig. (**4**)) organize and configure themselves to ensembles, which are the best available sets of sensors to execute that very sensing mission. If possible, according to the available sensor devices, a result for the recognition goal is returned to the requesting entity on top. During the execution of a sensing mission, the system reacts at runtime on topological changes (disconnects, connects or re-connects) in the sensor infrastructure. In the middle of the illustration we see a knowledge-base and knowledge-processing unit (the Ontology Processor), which is indispensable in an opportunistic system for (i) goal processing and translation, (ii) for describing semantic relations from the sensor's capabilities to the capabilities required for a sensing mission and (iii) to configure coordinated sensor ensembles.

4.3. Sensor Self-Description

An opportunistic activity and context recognition system does not specify at design time any recognition goal, initial sensor configuration, types of sensor systems that can be used in general or which hardware and software prerequisites the sensor nodes should be capable of. It also does not specify which type of data the sensors should deliver. Therefore, we have introduced the concept of sensor abstractions (Section 3.2) and we have explained the need for the sensors to be able to self-describe.

The sensor self-description is divided into two separate parts: (i) the technical and static description for a specific group of sensing devices (e.g. the physical characteristics of the aforementioned *Texas Instruments eZ430 Chronos*), and (ii) the dynamic and changeable description for every individual sensing and data- delivering entity. There are different reasons for this organization of the self-descriptions: firstly, we avoid possible dangerous redundancies in having more than one technical description for every single device and secondly we can reduce the effort to parse and reason through the documents. The risk of having redundancies is due to the fact that we might have several sensors of the same kind, no matter if they are of type *OnlineSensor*, *PhysicalSensor* or others (see Section 3.2 for a complete list), which play different roles within the same recognition goal. As an example, consider two accelerometers, available and ready to deliver data in the sensing environment. They are technically identical, the only difference is their location and their usage in the framework, as they might be used with different machine learning algorithms and for different

[8] http://www.sun.com/java/
[9] http://www.osgi.org

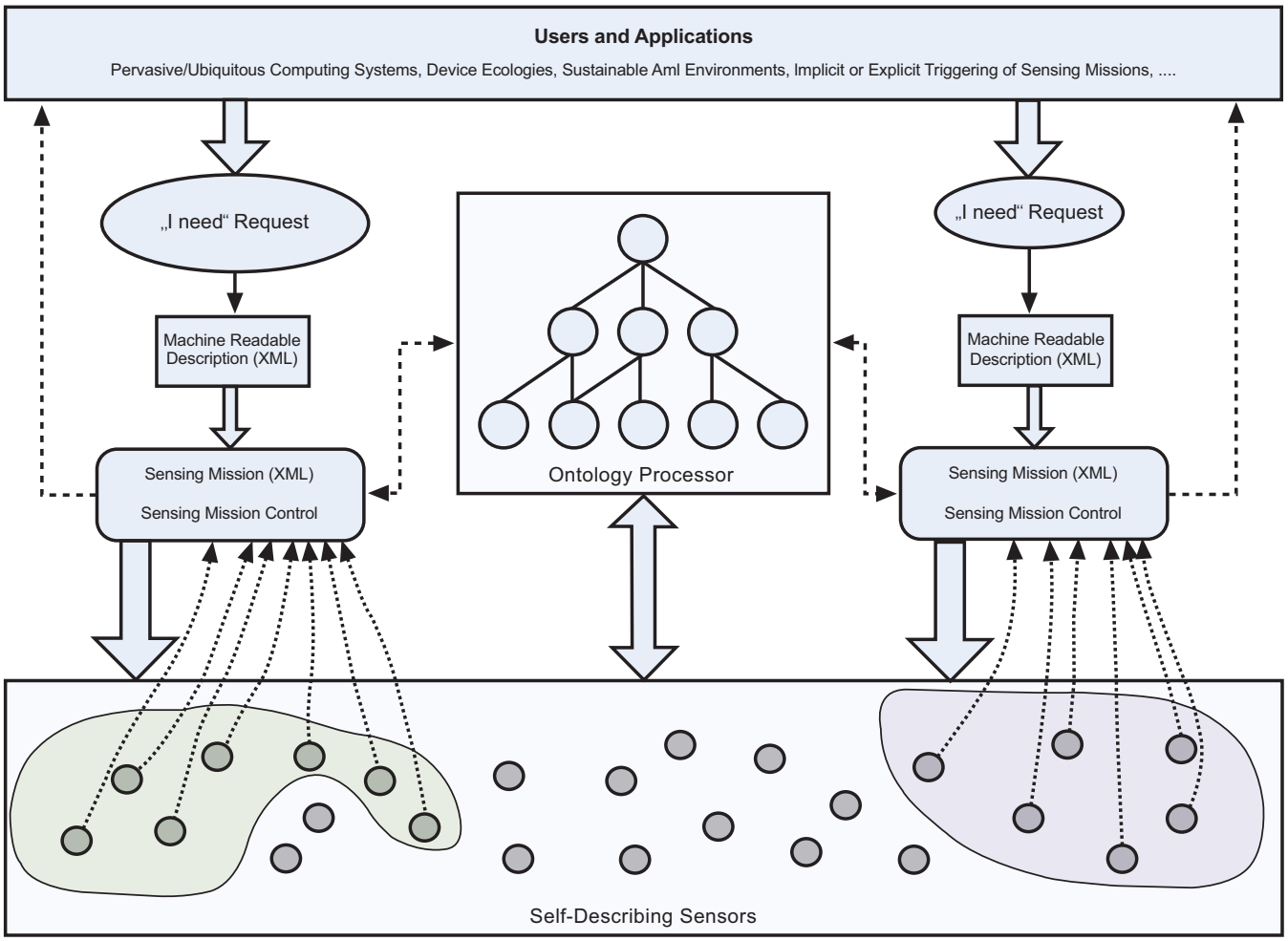

Fig. (4). The conceptual architecture of the OPPORTUNITY Framework [55]. At the top, users and/or applications can state a recognition goal (”I need” - Request) to the system at runtime. The goal is translated into a machine-readable form (the *Sensing Mission*). The self-describing sensors at the bottom are configured into sensing ensembles according to the sensing mission. The ontology in the middle enables the dynamic goal processing by providing a vocabulary of activities and therefore valid recognition goals and the semantic relations between them.

purposes (for example, one is attached to a fridge and can detect if the user opens or closes it and the other is installed on the user's foot to recognize when she is walking). Both are characterized by the same, unique technical self-description, but would need separate dynamic descriptions, which contain, among other things, their current location and their capabilities when a specified recognition chain is instanciated. Furthermore, the dynamic description for a certain sensor can also vary when the sensor is used for activity recognition in different domains (e.g. smart home or industrial application), without its technical characteristics being affected at all. This explains that it is reasonable to allow the number of dynamic self-descriptions per sensor to be variable and not pre-defined, since these depend on many factors, including the application at hand, whereas the technical self-description should be single and unique for an entire class of sensors.

The dynamic self-description follows the same standards as the technical description as it also uses the SensorML [52] scheme[10] and the according XML-namespaces as elements to build these XML-descriptions. On the syntactic level, the dynamic information is organized in two parts. A first block retains a value for what we call *Trust Indicator* (TI). This quality-of-service metric is a number in the range [0, 1] indicating how trustworthy the sensor data are at a precise moment (for more details, refer to 5.3). A second block is formed by the so-called *ExperienceItems*. Each ExperienceItem acts as snapshot to memorize the sensor capabilities in form of recognizable activities. The activities (represented by labels) are included in the system as part of the ontology and are stored as a subset in the ExperienceItem of a given sensor. Along each activity label, every ExperienceItem features a corresponding *Degree of Fulfillment* (DoF), which is a quality-of-service metric in the range [0, 1], which expresses how well a certain activity, is recognized (for more details, refer to 5.3). The ExperienceItem is used by the framework to configure an available sensor with the required

[10] http://schemas.opengis.net/sensorML/1.0.1/sensorML.xsd

machine learning algorithms and the correct training data to recognize a certain set of activities. In detail, the following information is stored in an ExperienceItem:

- *Method*: the assigned classification methodology is provided here. The framework uses this information to dynamically invoke the correct classifier.
- *Sensors*: defines the channels which have to be used for the configuration that is stored in this very ExperienceItem. If for example more than one sensor is listed in this section, a fusion-method can be easy defined which can be dynamically configured by the OPPORTUNITY Framework. Furthermore, the sensors section provides information about where the sensor has to be located to enable the recognition of the defined labels. This spatial information is also defined in the OPPORTUNITY ontology.
- *Labellist*: this list defines the activity classes that this sensor together with its recognition chain is able to recognize and to what extent (DoF for this class). The possible activity classes match the vocabulary of activities that are defined in the ontology.
- *Required Feature*: this part defines the required feature extraction method for this ExperienceItem. The OPPORTUNITY Framework uses this information to dynamically instantiate the recognition chain.

ExperienceItems build the connection point between the high-level ontological models and the low-level machine learning technologies. Further details on how the high-level framework capabilities are mapped to the machine learning algorithms are provided in Section 5, where the machine learning technologies and related opportunistic features are presented.

4.4. Goal Processing

In order to sense for a given goal, two parts have to be linked together as shown in Fig. (**6**). The upper left part shows the ontology that states all possible known goals within an application domain that the system can be asked to detect and their relations, and the goal tree that links the goals and the sub-goals inferred from the ontology (Section 4.5). Each node represents a possible recognition goal. The relations between the nodes symbolize how a goal can be split up into its sub-goals and therefore be detected by its sub-goals if e.g. no sensor is available that can detect the stated goal itself.

The right part in the figure shows the available self-describing sensors in the environment. Each sensor knows to which recognition goals it can contribute and is then selected accordingly. A recognition goal that is stated to the system has to be refined and translated and a sensor ensemble has to be configured accordingly to be able to contribute to the given goal.

First, the goal that has to be sensed for, expressed in the *goal description language* (see Section 3.3), is stated to the framework and analyzed. The given goal is checked against the goals stored in the ontology that builds the knowledge base of the *OPPORTUNITY Framework*, to determine if the given goal is known by the system and its components. Only if the goal is known in the ontology, it is a valid goal that can be understood by the framework and be further processed. In this case, the building of the sensing mission and the linking together of the resources needed to fulfill the mission in the best way are started and the possible sensor candidates are queried and configured to ensembles.

To do so, the available sensors are detected (set S). After the detection process, the search for a subset of the found available sensors that can satisfy the goal is initialized (set S_C). This is done by querying the *ExperienceItems* of the available sensors to find out if the sensor can satisfy the goal. The returned *candidate* set (S_C) is sorted according to the *DoF* and the *TI* of each sensor. Out of the selected candidates, the *"nBest"* sensors, are taken and put in the ensemble list. Each sensor in the ensemble (the set S_E) is then connected via a *wire* (a data pipe) to the sensing mission to be able to deliver data to the mission. After these steps, the ensemble is configured according to a given recognition goal and delivers data to the sensing mission. As the delivered data are only raw sensor data, machine learning algorithms that are described in the *ExperienceItems* have to be used to get a class output out of the data. How this recognition chain is set upon the ensemble is described in detail in Section 5.1.

The ensemble configuration is explained in more detail in the following part. An ensemble is the group- configuration of sensors that is best suited for executing a given sensing mission/recognition goal. The basic idea to realize this sensor clustering method is to first define a set of (available) sensors that could be used for a given sensing mission. This set includes possible candidates for the resulting ensemble. Based on the *capabilities* parts of the self-descriptions of the sensors, the ensembles can be structured. Last step in this ensemble compilation is executed by using the *configuration* -part of the descriptions. There the sensors are finally configured to an autonomous network. The following list describes the steps of the ensemble configuration in detail:

i) *Identifying Candidates*: This is probably the easiest part of the ensemble configuration process. A sensing mission that is gained out of an abstract goal contains machine-readable descriptions what shall be recognized. Based on these descriptions, this step simply reads and parses the *purposes*-part of the self-descriptions of the available sensor nodes to generate a set of possible candidates for the ensemble. The capabilities (how good is a sensor in executing a purpose, respectively a recognition goal) is yet not interesting. The set of possible candidates is used in the next step - *Ensemble Structuring* - to define the sensor nodes that will be really integrated in the ensemble. The available sensor nodes build the set S, the set of candidates is called S_C. The set of candidates is equal or a subset of all available sensors ($S_C \subseteq S$).

ii) *Ensemble Structuring*: This step targets the quality of the sensor nodes configured in the ensemble according to a recognition goal. A set of possible candidates has been gained in the previous step (*Identifying Candidates*). Within this step, the ensemble is structured with respect to the degree of fulfillment of (i) every single sensor node and (ii) the combinations of sensor nodes. Therefore, the *capabilities*-part in the

self-description is used, where a metric for every purpose that the sensor can be used for is provided. This value describes to which extent the sensor can execute a recognition goal. Based on these values the best set of sensors that is best suited for a given sensing mission can be configured to an ensemble. Let S be the set of sensors that are available and let S_E be the set of sensors that build the ensemble for a given sensing mission. S_E is a subset of S_C or equal to S_C ($S_E \subseteq S_C \subseteq S$). As already mentioned, the outcome is a set of (yet still not configured) available and connected sensor nodes that are best suited to execute a mission (S_E).

iii) *Ensemble Configuration*: The input for the last step is the set S_E that defines the sensors that are best suited for the sensing mission. Within this step the ensemble has to be configured, which includes network configurations to enable communication between the sensor nodes. Furthermore, it could be possible that one node is of type *OnlineSensor* where a connection to a web-service or another online resource has to be opened. Other operations within this step, which handle the configuration, can deal with inline code that is provided in the self-description and that has to be executed during ensemble configuration or stubs from objects that are necessary for a sensor node.

As the ensemble configuration is a highly dynamic process it has to immediately react on changes of the available sensor infrastructure during the runtime of the system. The *Sensing Mission* reacts on changes in the available sensor infrastructure. If there is a change (e.g. appearance, disappearance, . . .), the system reinitializes the ensemble (re-)configuration process and checks for the availability of sensors according to the sensing goal. After the available sensors are detected, they are structured again and the ensembles are rebuild accordingly.

Last step is to re-configure the new found ensemble to be used in the sensing mission. This process of reorganizing the ensembles always takes place if a change in the infrastructure is detected to ensure to always have the best fitting sensor configuration according to a sensing mission.

4.5. The OPPORTUNITY Ontology

The ontology builds the common shared knowledge of the opportunity framework. An ontology has the highest degree of semantic, meaning that every relation can be modeled using an ontology. As we have an *open world* assumption, and we do not exactly know which relations will have to be stored in the future, an ontology is the most flexible way to model the knowledge base of the OPPORTUNITY Framework. The ontology builds the semantic connection between the available, *self-describing* sensors in the environment and the goal tree that is built during the goal processing identifying which sensors can be used to satisfy a goal.

The used concept in the OPPORTUNITY Framework is to store the activities that are possible to be recognized and their relations. As only activities are modelled, and no information about the needed sensing infrastructure is stored in the ontology, it is possible, to add new sensors to the system in the future, that only have to state to which goals (activities) they can contribute to (must be a subset of the activities stored in the ontology) in their *self-description*. So the use of new emerging sensor hardware is easily possible and is not limited by the information stored in the ontology.

As stated only the activities and their relations are stored in the ontology. So far we have identified the relations *uses* and *canuse*. The relation *uses* indicates that the activity must be detected as a sub goal to fulfill its related higher level goal. The relation *canuse* indicates that the activity is a sub goal of a higher level goal but needs not necessarily be detected to fulfill the goal (but it would be nice to have it). The following examples explain the difference in more detail:

Take the goal `BREAKFAST` that can be detected either by using a sensor that can contribute to the goal `BREAKFAST`, or `BREAKFAST` can be split into its sub goals and a goal tree can be built saying that `BREAKFAST` can be detected by detecting its sub goals `SIT`, `EATING`, `DRINKING`, . . . As these relations are *uses*-relations, each of the sub goals must be detected to satisfy the goal `BREAKFAST`.

For the second example take the goal `LOCOMOTION` that can again either be detected using a sensor that can contribute to this goal or by splitting `LOCOMOTION` into its sub goals and build a goal tree that says `LOCOMOTION` can be detected by detecting its sub goals `WALK`, `SIT`, `RUN`, . . . As these relations are *canuse* - relations each of the sub goals (either one or more) can be used to satisfy the goal `LOCOMOTION`.

Fig. (**5**) illustrates the two goals used in the above example and their sub goals.

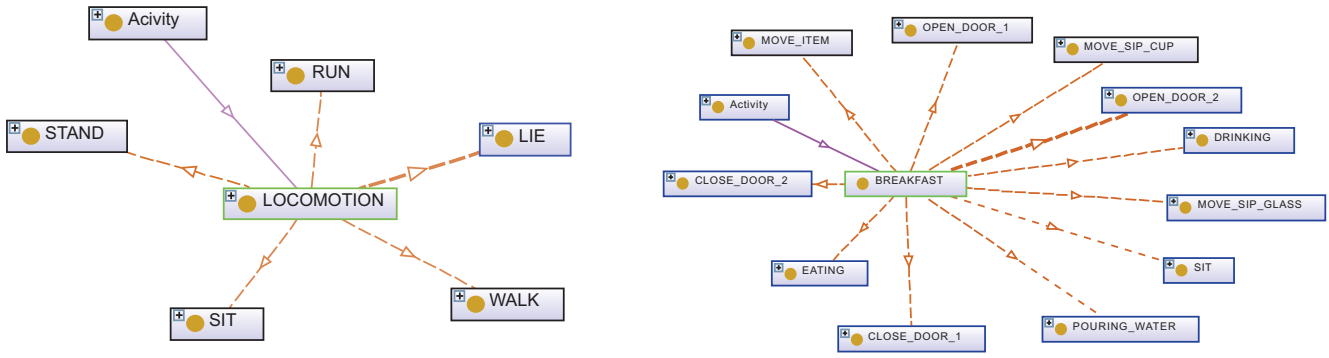

Fig. (5). Illustration of the activity `LOCOMOTION` and its related sub goals having a *canuse*-relation on the left hand side of the Figure the activity `BREAKFAST` and its related sub goals having a *uses*-relation on the right hand side of the Figure.

For the opportunistic approach in the framework it is extremely important to have the ability to know how an activity is related to other activities. This enables the selection of sensors that are not defined for contributing to the stated goal but instead can be used to sense a related sub-goal. So if all sub goals can be satisfied according to their relation in the ontology by invoking certain sensors, these sensors can be combined in an ensemble to fulfill the stated goal also if there is no sensor defined for the stated goal itself. This makes the opportunistic approach extremely flexible and powerful, in conjunction with an ontology that relates the activities.

5. INTEGRATION AND APPLICATION OF MACHINE-LEARNING TECHNOLOGIES IN THE OPPORTUNITY FRAMEWORK

5.1. Introduction

In the previous sections, we have shown how the construction of the ontology and the goal description language allow the framework to handle opportunistic configurations, where the resources are not statically defined at design time. To effectively translate this flexibility into framework operations, there is the need for a corresponding dynamic way of instantiating and adapting the machine learning tools, which are needed to fulfill the recognition goals, processed by the framework. The scenarios that we envision in terms of dynamics of the sensing infrastructure include:

- The appearance of a new sensor, capable of self-describing and pre-trained to contribute to a certain goal;
- The appearance of a new sensor which is still not able to contribute to a goal;
- The modification of the trust of a sensor, as it delivers faulty or anomalous data.
- The update of the sensor self-description to manage experience.

Fig. (**6**) depicts the interplay between the goal processing and reasoning engine, which define the sub-goals that can be recognized by the current sensor configurations and how they fit into the sensing mission, and the machine learning tools (recognition chains) which are used to provide the output needed by the mission. The link between these two "worlds" is provided by the sensor self-descriptions. These XML structures contain various pieces of information about the sensor, like the hardware, its location on the user's body or in the environment etc, but also have the crucial role of making the two following links: one to the ontology, advertising what goals or sub-goals they can contribute to and to which extent, the other to the recognition chains that are needed. The structures, which provides these links, are nodes in the XML self-description called *Experience Items*. These collect a piece of experience (hence the name), that is, the knowledge that a particular sensor or sensor set has been able to contribute to a certain sensing mission in the past and can then be used again for the same mission. Each *Experience Item* is a description of the recognition chain that should be associated to the sensor to fulfill one or more goals, along with the degree of fulfillment (DoF) with which each goal is achieved with that precise recognition chain. *Experience Items* contain also links to the training data which are needed either by a classifier or by a fusion block

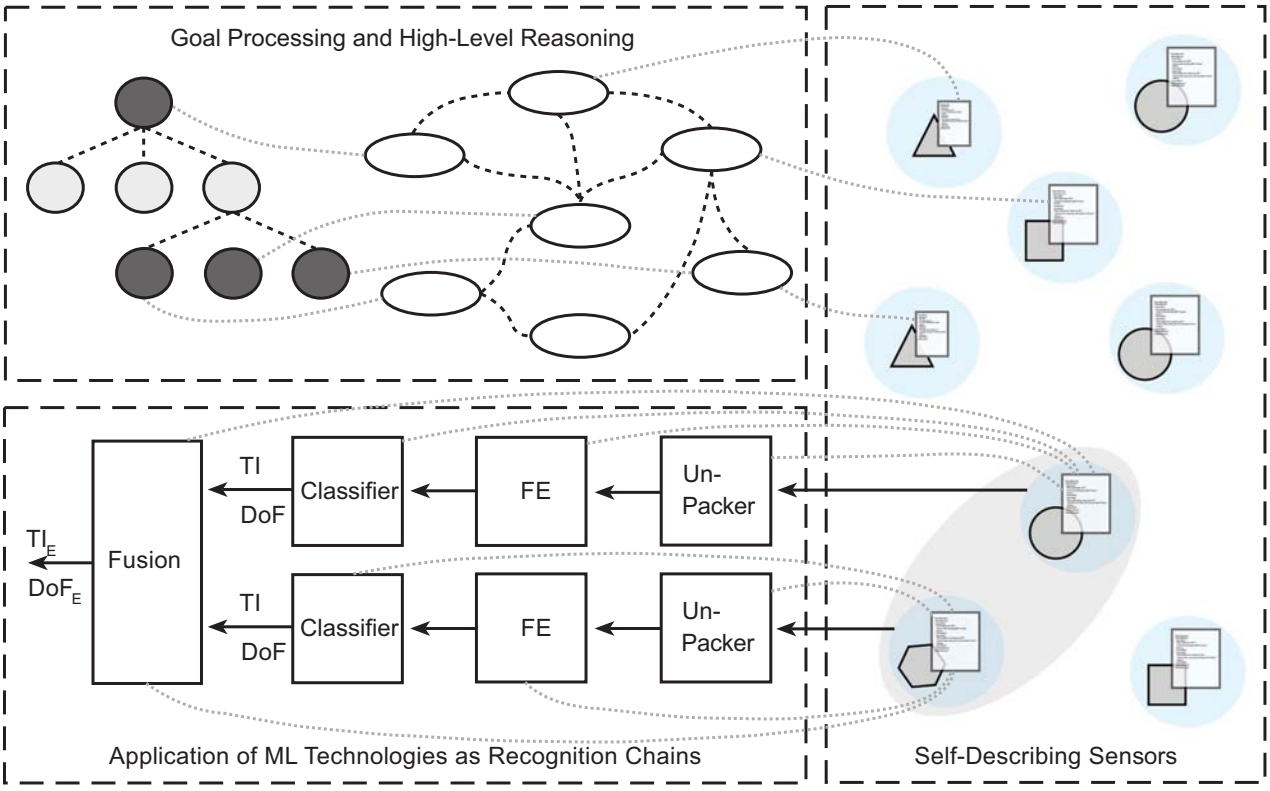

Fig. (6). This scheme illustrates the three main parts of opportunistic sensing that have to be interconnected, namely (**i**) the self-describing sensors, (**ii**) the machine learning technologies, and (**iii**) the high-level goal processing and reasoning capabilities.

(e.g. a Decision Template [61]). The following sections explain how the above mentioned dynamic changes in the sensing infrastructure are handled by the framework on the machine learning side and the role played by the self-descriptions and Experience Items in enabling this.

5.2. Transfer Learning

In an open-ended configuration, new sensors may be discovered in the environment or on the user's body and they might not possess a complete self-description, or they might not yet be trained to fulfill a given recognition goal. In the machine learning and ubiquitous computing literature, some solutions can be found that to a certain extent cope with such situations.

If at least one sensor is already present and trained data are available for the specific recognition goal, then we can employ transfer learning approaches [62]. In order to create new Experience Items for the new sensors, transfer learning approaches require that the sensors operate in the same domain (e.g. both measuring acceleration) and that they operate the same feature extraction, or that meta-features can be defined, which can be linked to both pre-existing and new sensors [63]. This is unfortunately not always the case, and from a framework point of view, we want to keep the methods as general as possible, so that they operate regardless of the specifics of the single sensor or of its recognition chain.

We therefore propose a transfer learning approach based on the assumption that there are sensors, which already can contribute to the sensing mission, meaning that there are Experience Items available for one or more sensors. The framework organizes these resources, which are then automatically connected to the sensing mission, using the ontology queries and goal interpretation described in 4.4 and 4.5. Furthermore, the new resource is connected to the sensing mission, but used as an observer, rather than a contributor. The activities recognized by the sensing mission are then used to incrementally train the new sensor's recognition chain (see [57] for more details on the framework implementation and [64] for more technical details on the machine learning side). A schematic representation of this mechanism is shown in Fig. (**7**). After every iteration of the incremental training done on the new sensor, its performance is estimated by trying to use it for the goal expressed by the sensing mission. The output delivered by the new sensor is compared with the output of the sensing mission and the rate of agreement is computed, as the number of times that the outputs are matching with respect to the total number of measurements. In this way, the DoF of the new sensor with its recognition chain is estimated. The outcomes of the transfer learning procedure in the framework are then a trained sensor and its DoF. These pieces of information are used to compile an Experience Item for the new sensor, which is then in turn able to participate to future sensing missions.

This approach is sufficiently general for our purposes, because it does not imply the existence of any relationship between the different sensor modalities, feature spaces etc, allowing then the framework to be used with many different sensor sets in an opportunistic way. Furthermore, the architecture of the framework and of the transfer learning do not pose constraints on the kind of sources of information which deliver results for a specific sensing mission. This means that many different, albeit heterogeneous sources can be combined to contribute to the same mission. For example, if the goal is detecting the activity `STAND`, even simple sensors like magnetic switches mounted in doors or other furniture items can contribute, by leveraging the assumption that a person is normally standing while opening or closing a door [23].

5.3. Anomaly Detection and Reconfiguration

According to the chosen sensing-mission, the framework selects the best set of sensors.

As the sensor configuration or sensing hardware can change during runtime of the system, the framework may change the configuration if the current one cannot fulfill the mission. The applied machine learning methods have to support the modification of the sources. The best architecture of classification that can cope with sensor addition/removal and degradation is the classifier fusion [61, 65]. In such fusion method, a classifier is assigned to a subset of sensors (e.g. the onbody sensors which are located at the same physical location), and each classifier makes a decision independent of other classifiers. This architecture allows to remove a faulty/degraded sensor or add a new sensor to the

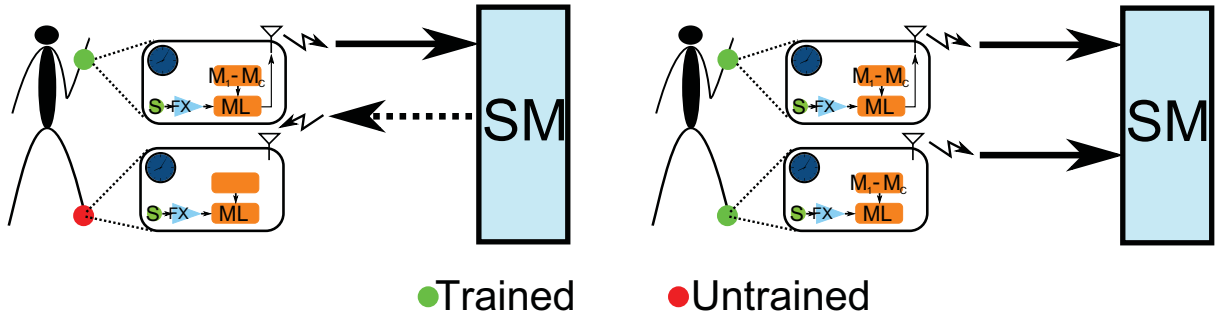

Fig. (7). Schematic description of the transfer learning on an example setup. Initially (left side), a trained sensor (S) along with its feature extraction block (F) and stored classifier models $M_1 \ldots M_C$, is placed on the right upper arm and contributes (solid arrow) to the sensing mission. A second sensor on the shoe does not advertise any experience items for that goal and is then only *observing* (dashed arrow) the sensing mission. While observing, the new sensor gather labels and associates them to the corresponding measured signals, incrementally training a classifier. Finally (right side), it can contribute to the sensing mission as well, having now an Experience Item.

network without the need of re-training or manipulating other classifiers with the new configuration.

Each ensemble of classifiers (provided as *Experience Item* in the framework) which is assigned to a fusion provides a set of values called Degree of Fulfillment (DoF) indicating how well the ensemble can recognize a mission (class). Note that the ensemble can also consist of only one sensor/classifier. The DoF values are between 0 and 1 where higher value shows better recognition rate. DoFs are estimated at the training phase of each ensemble and it can be based on the accuracy of classification, e.g. F-Measure [66], or a function of the accuracies of each classifier for that mission.

In opportunistic sensor networks, there is a possibility of changes or degradation in the sensor reading because of displacement, low power, environment fluctuations, and etc.. The other type of change regards the change in the behavior of a sensor with respect to the others, i.e. a sensor is not in harmony with other sensors as before. Hereafter, we call all these type of degradations as *anomalies*. If the network is able to detect such anomalies it is possible to rectify or remove them from the network, collaborating with the framework to reconfigure the ensemble of sensors to keep the performance of the system as high as possible. Sagha *et al.* proposed a method to handle temporary disconnection of sensors which leads to the missing data in the framework by imposing expected values in the classifier fusion [67]. In [68], we proposed an approach to detect anomalies in the network. The anomalies are detected when a classifier is not working in harmony with other classifiers in the ensemble. Fig. (**8a**) shows the accuracy of detecting anomalous sensors based on the proposed approach when the ensemble consists of seven on-body sensors.

The anomaly detection provides values for each classifier in the ensemble indicating the reliability of each one at any time, namely Trust Indicator (TI value). Fig. (**8b**) shows the TI values of the sensors in an ensemble and its decrement when a sensor is transmitting abnormal data. The combination of these values gives the level of trust for the whole ensemble, TI_E. This value is different from DoF, so that TI_E indicates the online reliability of ensemble, while DoF is static and shows the initial recognition rate of the ensemble for a mission. Whenever the detection algorithm detects anomalies it updates the TI_E value, giving the framework a measure of how well the current configuration can fulfill the mission. In [56], we combined TI and DoF values together and once the measure is less than a threshold (obtained from the next best ensemble/*Experience Item*), the framework changes the configuration of the sensors by looking at the available *Experience Items* for the current mission, and selects the next best one.

The collaboration between classifier fusion, anomaly detection and reconfiguration (by means of DoF and TI) gives the ability to the framework to select the best ensemble at any time whenever a sensor fails or degraded or is not behaving as before. This provides a long life classification, robust against changes in the network.

5.4. Sensor Location Self-Description

In opportunistic systems the user is free to place the sensors wherever he wishes on body. Having the sensors autonomously discover their on-body location is a crucial feature especially with respect to the use of smart phone platforms for activity recognition. We have developed a method that allows motion sensors to autonomously find out on what body part they are positioned and include this information in the sensor self description [69, 70]. This approach is designed for acceleration sensors.

An overview of the method is shown in Fig. (**9**). We first distinguish between time periods where the user is moving and those where he is not. The stationary segments are used for calibration. In the motion segments we then apply a three stage methodology consisting of feature computation, HMM based frame by frame recognition and particle filter based smoothing:

Feature Calculation

Feature extraction is done on a 1 sec. sliding window (0.5 sec overlapping). We only perform feature extraction on segments with enough activity. If the variance of a segment

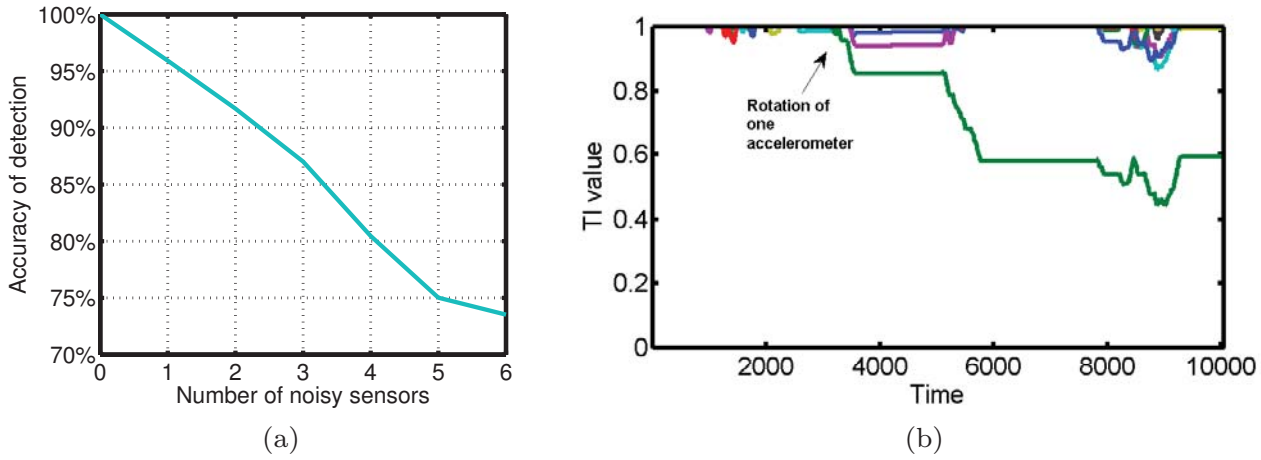

Fig. (8). (**a**) Accuracy of detecting noisy sensors in an ensemble with respect to the number of affected sensors. (**b**) Decrease in TI value of a sensor upon detecting it as anomalous.

on each axis tends towards zero and the magnitude towards 9.81 m/s2, we assume this is the gravity vector. To account for sensor shifts and displacements within a body part, we perform a constant orientation calibration for one axis, as described by Mizell [71]. We perform feature extraction on this vertical axis and the norm vector of the two vectors orthogonal to gravity, if not indicated otherwise with the feature. As an additional feature we also use the length of the last calibration/ rest period. For the acceleration sensors we calculate (1) the standard deviation and mean, (2) the sum of the norm of the differences in variance for the normalized axes divided by the variance of norm of the acceleration vector $\left(\frac{|\mathrm{var}(X_n)-\mathrm{var}(Y_n)|+|\mathrm{var}(X_n)-\mathrm{var}(Z_n)|+...}{\mathrm{var}(\mathrm{norm})} \right)$, (3) the number of peaks in the absolute value of the three axis using hill climbing with a threshold, (4) the fft center of mass, and (5) time of the last rest period.

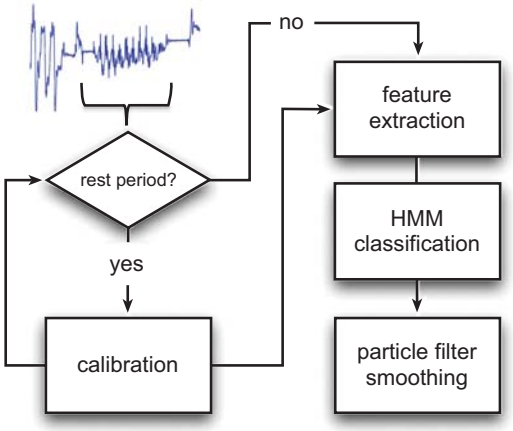

Fig. (9). Overview of the method for acceleration sensors to self-locate on the body. Once the location is found with a classifier and filtered to enhance accuracy with a particle filter, this information is advertised into the sensor self-description which can then be used by the OPPORTUNITY framework.

Hidden Markov Models

The features are calculated as described above and a sequence of 5 min. feature segments are feed into continuous HMMs. We use mixture of 3-5 Gaussian distributions to estimate the HMM output probabilities. We train a separate HMM for each body placement. Each HMM in itself is fully connected. Depending on the placement with different numbers of hidden states (hand: 5-6, torso: 4, leg: 5, head: 4). Training of each HMM is done by expectation maximization using the Baum-Welch algorithm. For evaluation, we feed the test sequence in each placement specific HMM and assign the classification from the HMM with the highest probability. On top of the HMM only classification, we use a majority decision window of size 10.

HMMs with Particle Filter Smooting

We input the 45 sec. sliding window hmm classifications as observations into a particle filter. The particles are initilized distributed equally with the placements of the data set to evaluate. The prediction model has a bias on not changing the placement classification; the probability for keeping the location class is set to 95 % steady. To obtain a classification we use the rounded mean over all particles (taking the weights into account). As seen in Fig. (**10**), reasonable results can be achieved with around 60 particles.

Example Results

An example of the method performance is shown in Fig. (**10**) for the data set that we use for our demonstration in section 7. It can be seen that when a device is placed at a new body location after on average 2 to 3 min the system "settles" on the new location and remains reasonably stable. At this point the sensor self-description is updated with the inferred location. The framework receives a notification of this and can assess whether the current recognition goal may make use of this newly self-characterized sensor. In some instance the system may falsely think that the device has been moved again. This is due to unusual motion sequences. The trust indicator is used to reflect this.

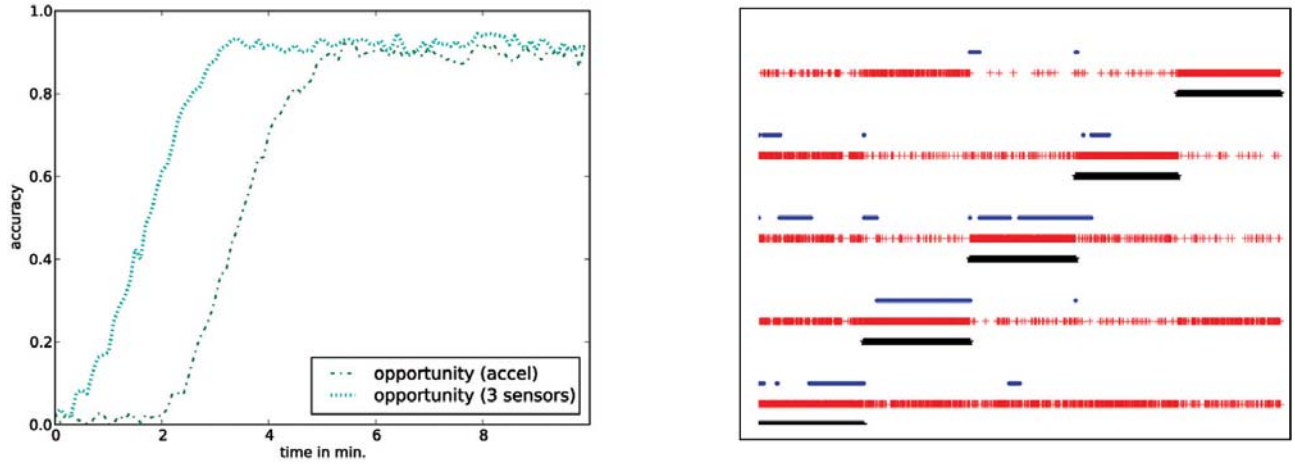

Fig. (10). Results the application of our method to the data used for system demonstration in section 7. Left: Accuracy of the recognition as a function of time after a device has been moved from one location to another. Results is averaged over 100 segments, each 10 min long using 60 particles filter. Right: Scatter plot depicting the performance of the HMMs with and without particle filter smoothing. Here 20 min segments are taken from the accelerometer only data, with 45 sec. HMM classification (around 59 % correct). With a particle filter with 40 particles the localization is 78 % correct.

6. OPPORTUNISTIC SYSTEM DESIGN TOOL CHAIN

In the previous sections, we have discussed properties of the framework in its runtime behavior. The successful operation of the framework is also connected to the quality of the data, which are used to provide the training models linked within the Experience Items. Prototyping and providing Experience Items involves a set of time consuming steps starting with data recording, through data cleanup, annotation, iterative system improvement to final testing. In opportunistic systems, where we need to optimize and test the systems reaction not just to a single sensor setup, but its ability to deal with sensor configuration changes those steps are even more laborious.

We have developed a toolchain that seamlessly interfaces with the OPPORTUNITY framework to sup- port data collection, enrichment, management and its use for defining the system's Experience Items. The toolchain addresses the needs to scale up the size of datasets used for the design of opportunistic activity recognition systems. The toolchain allows to acquire large datasets of a wide range of heterogeneous sensors. It provides means to "crowd source" the data aquisition and annotation, and to share the resulting datasets. It has been extensively used to aquire the dataset used in the OPPORTUNITY framework demonstration in Section 7. In this setup it was used to crowd-source the annotation 140 GB of sensor data and video footage corresponding to 25 hours of data recorded from 12 subjects and 72 sensors and generate the corresponding Experience Items [17].

The design toolchain consists of three integrated phases (see Fig. **11**). The first phase is the *data collection* phase where training data is recorded from real sensors and video cameras during various experiments. During this process the status of all sensors can be monitored to ensure the quality of the recordings. The recorded data sets are stored in a central database, which is maintained in the *data management and annotation enrichment* phase. Data sets can be reviewed, organized, and re-labeled using graphical user interfaces. Furthermore, specific reproducible data traces can be generated from the database, which are necessary for training and evaluating methods for context recognition. In the *generation of Experience Items* phase the training data is used to parameterize the context recognition methods, define the Experience Items and the sensor static self-description. For more details on the tool chain see [72].

6.1. Data Collection

For large-scale activity dataset recordings we combine a data acquisition toolbox (CRN Toolbox) designed for recording of multimodal sensor data streams running on desktop computers [39] and a mobile data recording tool for the Apple IOS and the Google Android platforms (ContextLogger) which supports the crowd-sourcing of data acquisition. The ContextLogger is available through Apple's AppStore and Google's Android Market. The application allows for an easy recording and labeling of all sensors on the device. These include accelerometers, gyroscopes, compass, GPS, sound, WiFi information (name, MAC address, signal strength, protection, visibility), and proximity sensors. The recordings can be uploaded directly to the Context Database described below.

In order to guarantee signal integrity during the design phase we developed *MASS* (Monitoring Application for Sensor Systems) that helps monitoring and documenting experiments with multimodal sensor setups. It features graphical and tabular views for visualizing sensor uptimes and dynamic plots of live sensor signals for quick checking of signal quality. MASS automatically finds available sensors from instances of the Context Logger and the CRN Toolbox.

6.2. Data Management and Annotation Enrichment

Once sensor datasets are recorded they need to be maintained and prepared for use in the context recognition methods. Another important aspect is the ease of sharing data across different user groups, to handle the issue of large dataset annotation through crowd-sourcing.

Datasets are stored in the *Context Database* running Apache CouchDB, which offers good scalability and support for replication. Users may easily replicate parts of the data-

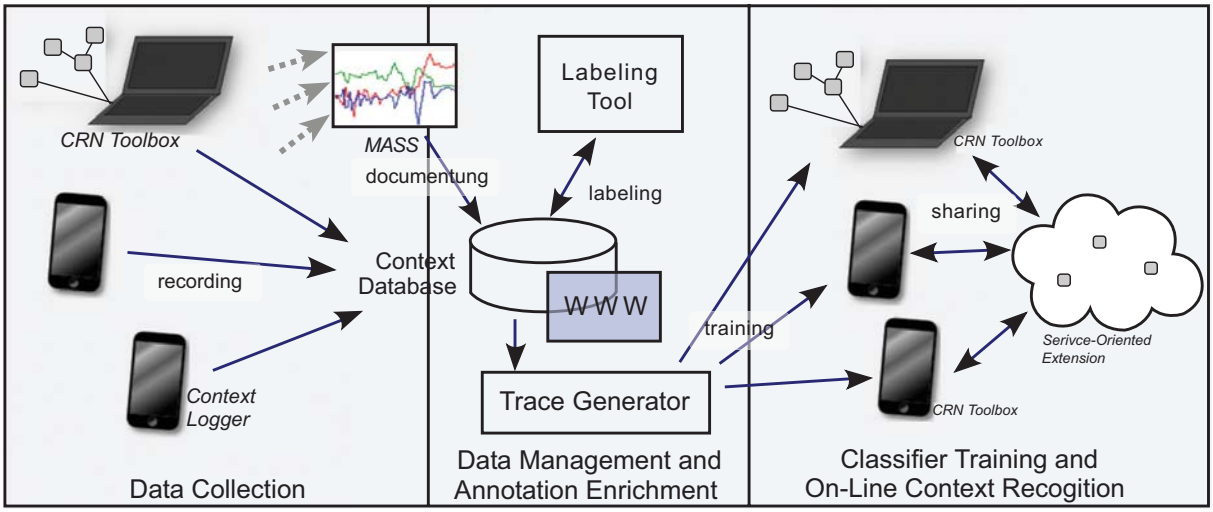

Fig. (11). Overview of the tool chain for opportunistic data collection and system design.

base locally, e.g. for performance reasons, and still continue to work on exactly the same interface. Users may upload recordings using the Context Logger on IOS and Android devices or the CRN Toolbox, and review the recordings and enhance their annotations using the Labeling Tool.

We developed the *Labeling Tool* for dataset annotation and graphical exploration of synchronized video footage and sensor data streams. It allows crowd-sourcing of the annotation effort by splitting annotations on multiple independent and fully configurable "tracks". It integrates with the tool chain by connecting to the Context Database for loading and saving the data sets, and by supporting among others the format of data files recorded with the CRN Toolbox.

In order to reduce information loss inherent when transforming sensor data into the form needed by machine learning methods, we store only raw, unmodified sensor data in the Context Database. As many machine learning methods have very specific needs for the format and properties of their input data we developed the *Trace Generator* which transforms raw sensor data from the database into specific data traces for each method.

6.3. Generation of Experience Items

The labeling tool allows to export data sets into flat, synchronized files for further exploration in tools like WEKA, Scipy, and Matlab. These tools are typically used to design the Experience Items of the system by defining features and classifiers to use and their parameters.

7. SYSTEM DEMONSTRATION

7.1. Introduction

In this Section we demonstrate how the two "worlds" of high-level frameworks and low-level machine learning technologies work together. Therefore we show and demonstrate four features that emphasize the opportunistic requirements of an activity and context recognition system:

(i) Querying of sensors according to a recognition goal and configuration of a single-/multi-sensor ensemble,

(ii) Sensor appearing and ensemble (re-)configuration,

(iii) Transfer of recognition capabilities from a trained sensor to an untrained source, and

(iv) Detecting anomalies of the sensor datastream (e.g. due to sensor rotations or sensor displacements) and accordingly reacting to further fulfill the recognition goal.

For this purpose, we use a large scale dataset collected to exhibit the sensor-rich characteristics of opportunistic sensing systems (Fig. **12**). The dataset comprises 25 hours of activities of daily living, collected from 12 subjects. It contains the data of 72 sensors of 10 modalities and part 15 networked sensor systems deployed in objects, on body and in the environment. This activity of daily living scenario and the dataset is used to develop, evaluate and test opportunistic technologies [17, 64], together with the aforementioned features.

In the following Sections 7.2 to 7.5 we explain the operation of the OPPORTUNITY ecosystem. We emphasize the interaction between the framework and the data processing in situations illustrative of typical modes of operation of the opportunistic activity recognition systems.

7.2. Goal Querying and Sensor Configuration

As shown in Fig. (**13**), when a goal, in this case `WALK`, is stated to the OPPORTUNITY Framework it queries the available sensors (illustrated by the green arrows) to get knowledge about which sensors can contribute to the given goal. Each sensor that can contribute to the goal, as this knowledge is stored by each sensor in its *self-description*, responds how good it can contribute to the goal in respect to the *DoF* and the *TI*. The OPPORTUNITY Framework analysis the responds from the sensors and configures an ensemble, that's the best set of available sensors to contribute to the given goal (illustrated by the yellow arrow).

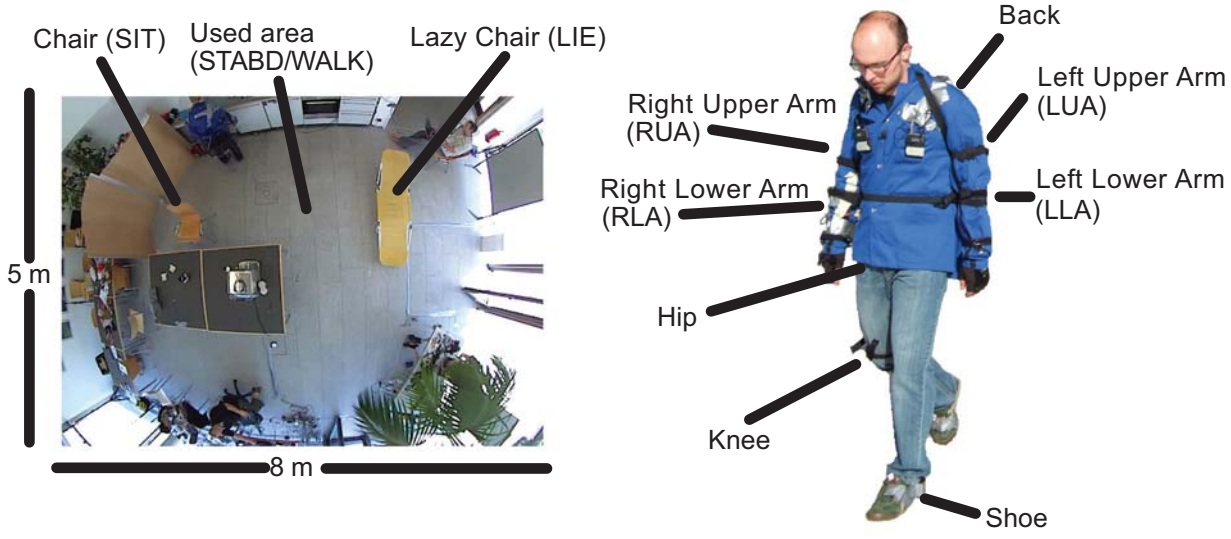

Fig. (12). This Figure shows the real world setup in a kitchen scenario where we placed 72 sensor with 10 modalities in the environment, on objects and on the body (see [17] and [64] for further details).

7.3. Sensor Appearing and Ensemble Configuration

As shown in Fig. (**14**), if a sensor appears, meaning that its presence is getting known by the OPPORTUNITY Framework, the sensor is first queried to get its capabilities (indicated by the green arrow). After the response of the sensor, the OPPORTUNITY Framework knows to which goals and to which extent the sensor can contribute to. If the

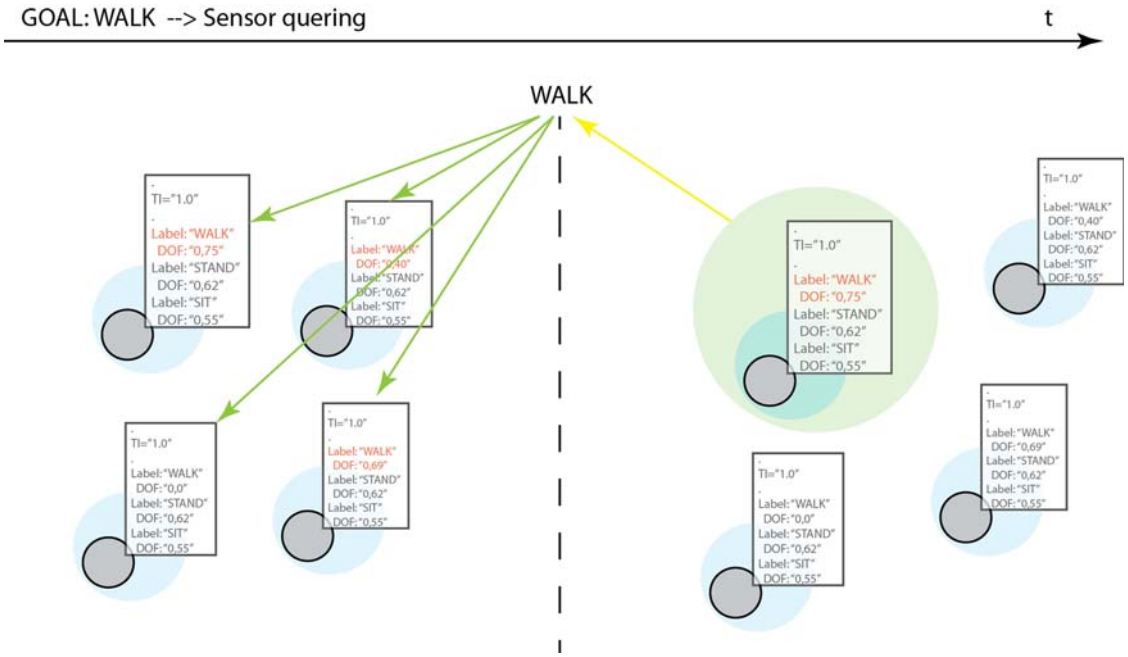

Fig. (13). Illustration of the querying of the available sensors by the OPPORTUNITY Framework to get knowledge about which sensors are capable of contributing to the goal WALK. The available sensors that are capable of contributing to this goal (according to their self descriptions) respond to the query and the OPPORTUNITY Framework selects the best ensemble that can contribute to the goal. The time line goes from the left to the right, as the green arrows on the right side indicate the querying of the available sensors, and the yellow arrow on the left side indicates the delivering of data to the given goal by the selected (single sensor-) ensemble.

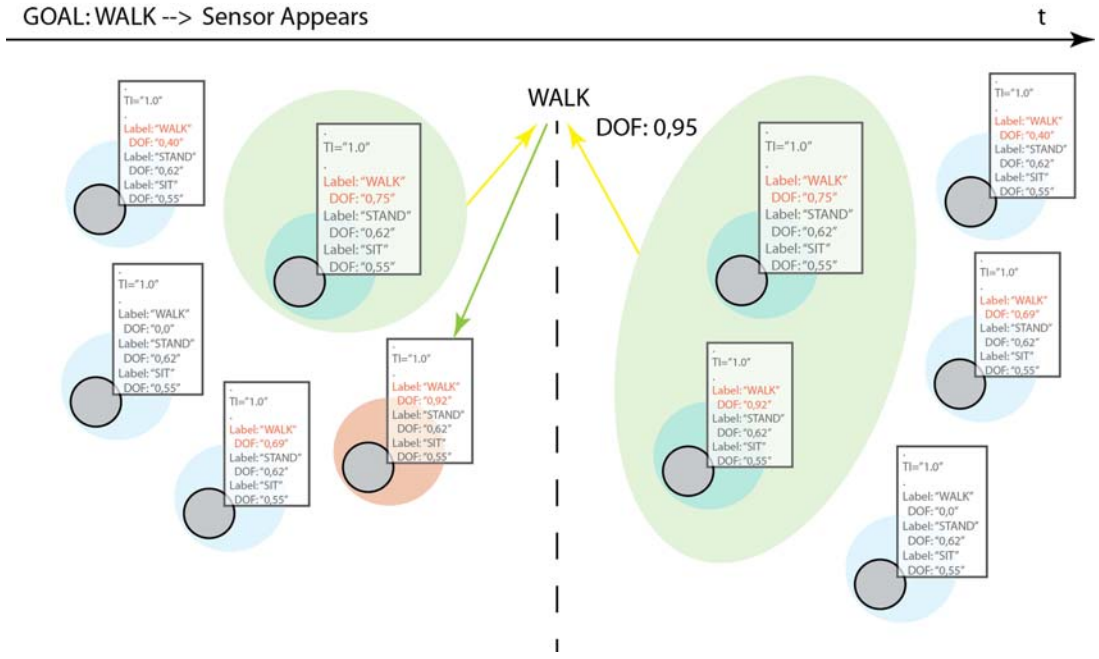

Fig. (14). Illustration of a sensor appearing that can be used in an ensemble to contribute to the goal WALK. Therefore the OPPORTUNITY Framework recognizes the appearance of the sensor and queries its capabilities (indicated by the green arrow). The sensor responds that it can be used to detect the goal WALK. So the OPPORTUNITY Framework reconfigures the current ensemble as this sensor can be used in conjunction with an other to contribute to the goal WALK with a higher DoF as any single sensor. The green ellipse surrounds the sensors that are now combined in an ensemble and the yellow arrow indicates the contribution of this ensemble to the given goal.

sensor can contribute to the actual goal the framework analysis if the use of this sensor would improve the recognition of the actual goal. If this is the case the ensemble is restructured and the sensor is used either alone or in conjunction with others to get the best recognition result. In the example in Fig. (**14**) the sensor is combined with a second one using fusion techniques (Section 5.3) as this combination has a higher recognition rate as any available single sensor.

7.4. Transfer of Recognition Capabilities

As shown in Fig. (**15**) the OPPORTUNITY Framework is capable of doing transfer learning from one sensor to another (Section 5.2). This is useful if one sensor is known to be a candidate to contribute to a goal but it is not yet trained for it. To indicate that a sensor is a candidate for a goal, it has to have a label in its self-description with a *DoF* of 0.0. If the OPPORTUNITY Framework detects the appearance of a sensor it queries this sensor. If it detects a label in the self-description of the sensor corresponding to the actual recognition goal with a *DoF* value of 0.0 it knows that this sensor is a learning candidate for the actual recognition goal as shown on the left side of Fig. (**15**) (in our case the label WALK with a *DoF* of 0.0). The OPPORTUNITY Framework reacts with connecting this sensor to the actual sensing mission and trains this sensor respectively its recognition chain using transfer learning with the labels that are produced by the current sensing ensemble as shown in the middle sketch of Fig. (**15**). During the training process the *DoF* of the trained sensor increases according to the comparison of the output of the teacher and the learner that is done periodically after certain amount of time to get a reliable value for the *DoF* of the learner. After the learning process the trained sensor can be used to detect the goal that it was trained for by the ensemble (WALK) indicated by the *DoF* of that label (in our case WALK).

7.5. Anomaly Detection and Ensemble Reconfiguration

As sensors can deliver faulty data due to packet loss during the physical transmission of the data or on a change in position, rotation or malfunctioning of the sensor itself, each sensor is aware of how plausible the data it delivers is. This reliability measurement named *Trust Indicator* indicates the self-awareness capabilities of the sensors thus describing the plausibility of data delivered from the sensor. Changes in the *TI* value influence the *DoF* as when the quality of the

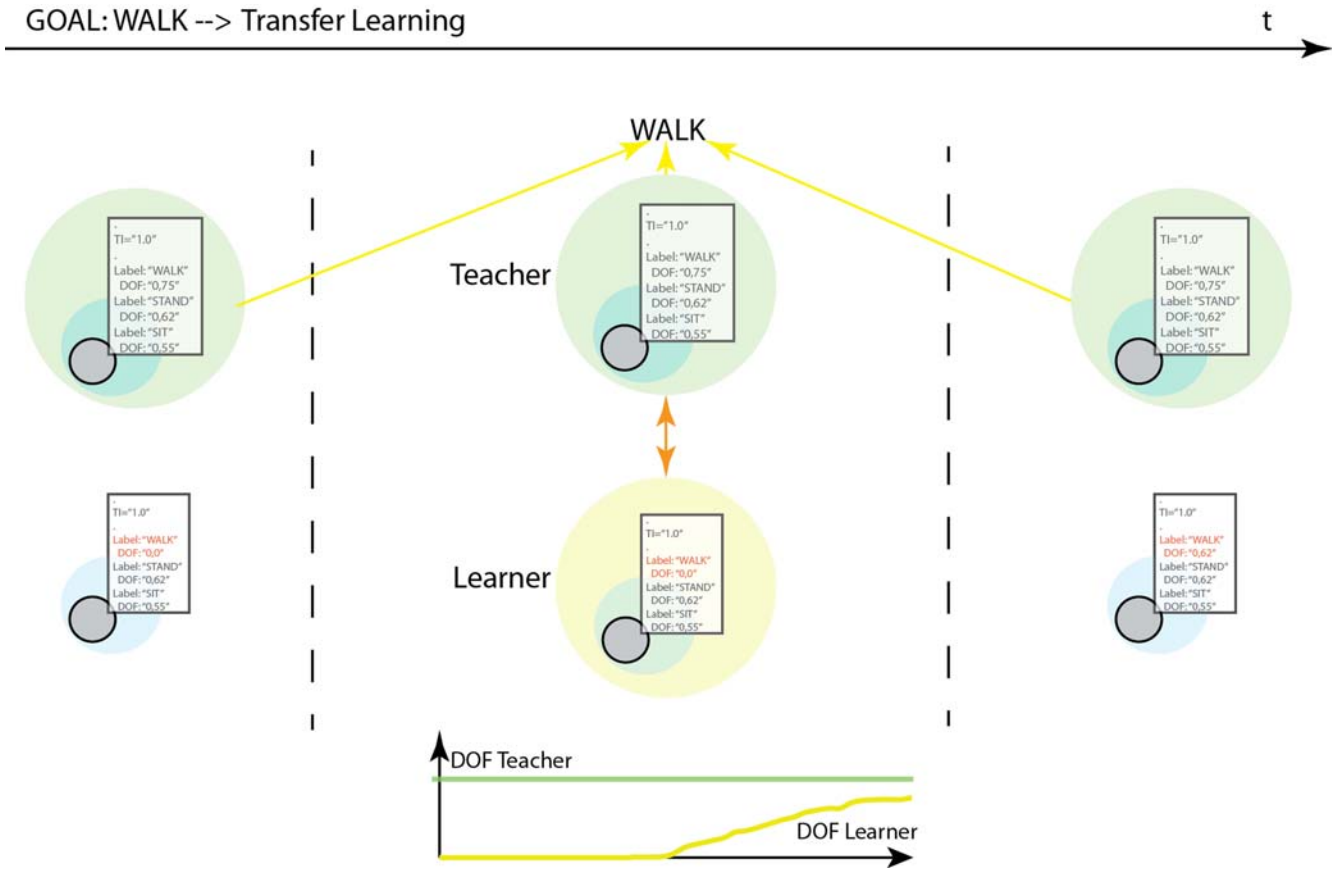

Fig. (15). The OPPORTUNITY Framework detects a sensor that can be trained for the goal WALK. This is done by querying each appearing sensor if its self-description contains an entry with a label corresponding to the actual recognition goal and a DoF value of 0.0. If such a sensor is found it is connected to the current running sensing mission and trained using transfer learning with the labels that are generated by the current mission. As shown on the left, there is an ensemble with one sensor contributing to the mission WALK. The OPPORTUNITY Framework recognized a second sensor that can be trained for the goal WALK (DoF for *WALK = 0.0*) and connects it to the current ensemble as shown in the middle frame. After the training that is sketched on the right, the aforementioned sensor is trained for the goal WALK as indicated by the new DoF of 0.62 for the goal WALK.

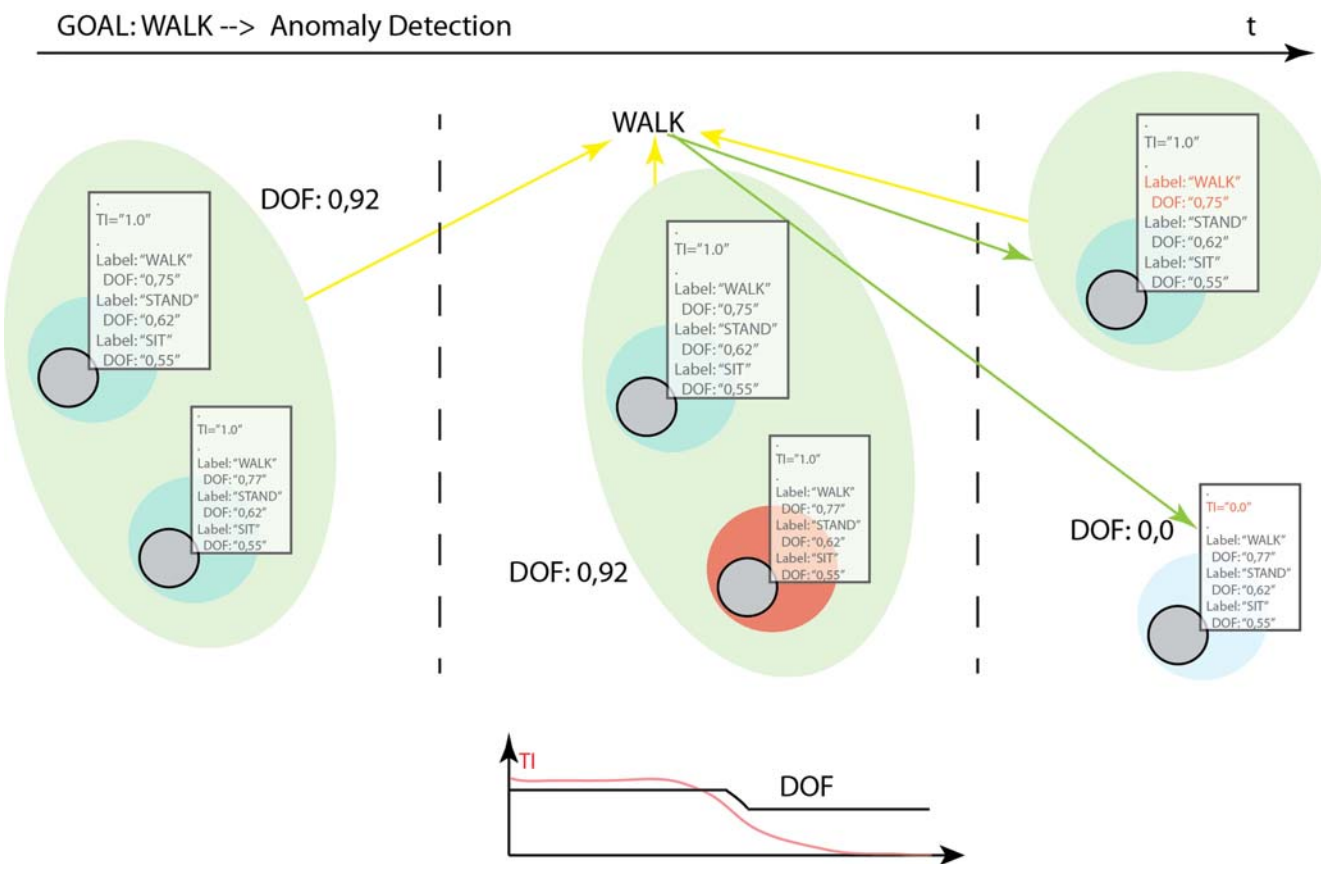

Fig. (16). Illustration of a faulty sensor which is detected by the OPPORTUNITY Framework using anomaly detection and therefore the ensemble is reconfigured. The left side shows an ensemble of two sensors contributing to the goal WALK. As one sensor is detected by the OPPORTUNITY Framework to deliver faulty data (indicated by coloring the sensor red), as shown in the middle sketch, its TI is decreased accordingly. As the TI value of this faulty sensor falls below a certain threshold, the OPPORTUNITY Framework reconfigures the ensemble as shown on the right side and uses the remaining single sensor as ensemble as it has a higher DoF as the faulty one for the goal WALK.

delivered data changes, also the *DoF* changes. As the *DoF* is a fixed value that is generated during the learning process of the sensor/classifier using performance measurements (e.g. insertions, deletions, confusion matrix, . . .) it is therefore multiplied by the TI to get the overall QoS indicator. As the TI value changes during the runtime of the system, the framework has to monitor the TI of each sensor and react on changes. Reacting means that if the TI, the plausibility of the delivered data, falls e.g. beneath a certain threshold, the sensor might not be useful for the sensing mission any more and thus a restructuring of the ensemble is necessary. Also when the TI of a sensor increases the framework has to react as this can lead to a reuse of the sensor in an ensemble. As shown in Fig. (**16**) on the left side there is an ensemble composed of two sensors contributing to the goal WALK. One sensor is detected to deliver anomaly data (Section 5.3) as shown in the middle sketch indicated by colouring the sensor red. As the TI value drops below a certain threshold the OPPORTUNITY Framework reconfigures the ensemble and queries for the most capable sensor as shown in the right sketch. The TI value of the faulty sensor has dropped to 0.0 so its DoF for the goals it can contribute to doesn't matter any more because the TI value is multiplied with the DoF to get the overall QoS indicator and so it is "0" and the sensor is not selected to contribute to the mission any more. The remaining sensor is therefore selected as the ensemble to contribute to the mission. This of course means a drop in the DoF value because this single sensor is not as good as the combined two before. But as one sensor of the before combined two delivered faulty data it was better to reconfigure the ensemble accordingly as to use a faulty sensor. So the OPPORTUNITY Framework is capable of detecting faulty sensors and reacts in reconfiguring the ensemble accordingly.

8. CONCLUSION AND OUTLOOK

Sensors providing information relevant to detect human activities are embedded in ever more everyday artifacts (e.g. clothing, furniture, cameras, smartphones, . . .). This motivates a shift away from deploying "application-specific" sensors to using sensors "opportunistically discovered" around the user for activity recognition.

The central contribution of this work is to propose a way to do human activity recognition with opportunistic sensing, despite the lack of guarantee about sensor kind, placement and availability at run-time. The potential benefits include higher system robustness, more comfort for users of activity recognition systems by relaxing constraints on sensor placement, and a recognition system that is not restricted to a specifically created ambient intelligence environment. The

proposed approach for opportunistic activity and context recognition is the OPPORTUNITY Framework and Data Processing Ecosystem. OPPORTUNITY comprises a sensing/context framework coordinating sensor recruitment according to a high level recognition goal. A matching between sensor node self-description and a goal representations allows to identify sensors contributing to the recognition goal. OPPORTUNITY then makes a dynamic instanciation of data processing elements to infer activities according to Experience Items. Experience Items indicate, for each sensor or sets of sensors, their capability to recognize a set of activities. They store the corresponding signal processing and machine learning parameters, such as features or classifiers used. Finally, the framework and data processing components of OPPORTUNITY are tightly integrated. This allows the framework to autonomously discover new knowledge about sensors at run-time thanks to novel data processing techniques. This knowledge is stored in Experience Items or sensor self-description. It expands the capability of the framework to use sensors for activity recognition, which is a key for operating in open-ended environments.

The framework is a flexible runtime environment based on an OSGi implementation. To cope with the heterogeneity of a plethora of emergent sensors, we introduced the concept of sensor abstractions that provide a common accessible interface from different sources of environmental data including physical devices and immaterial data-sources, like online webservices. Among available resources, the sensors effectively used are defined by a dynamically stated recognition goal. Thus, the framework retrieves the candidate sensors in a top-down fashion, which is reversed compared to traditional approaches. To support this, we define a syntax, based on SensorML, that allows the sensors to self-describe their characteristics. Sensor self-descriptions contain (i) a static technical description and (ii) a dynamic description. The latter expresses the sensor's recognition capabilities when specific data processing methods are used and the trustworthiness of the data, which the sensor is delivering. *Experience Items* reside in the dynamic part and are a key novelty of this opportunistic approach. They are fragments in the self-description that build the connection point between the high-level semantic modeling and the low-level machine-learning technologies. In order to support the dynamic formulation of recognition goals at runtime we introduced a goal language and processing capabilities by using a knowledge base in form of an ontology. We showed how the framework operates in situations typical of opportunistic activity recognition: (i) Goal querying and sensor configuration, (ii) managing ensemble configuration upon sensor appearance, (iii) autonomous transfer of recognition capabilities to unknown new sensors, (iv) autonomous detection of anomalies in the sensor datastream, and (v) autonomous discovery of on-body sensor placement.

In this work we introduced the notion of *Degree of Fullfillment* (DoF). It expresses how accurately a certain sensor set can contribute to a sensing mission. To enable better scalability and flexibility, future work could investigate how to automatically estimate how the DoF varies in function of the sets of sensors available. Similarly, we introduced a *Trust Indicator* (TI) to measure how reliable the data provided by a certain sensor is. More efficient methods can be developed to estimate TI values and different ways of combining single TIs into an ensemble TI should be evaluated systematically. Information theoretical approaches may support this by quantifying the contribution of each sensors to a recognition task in bits. Recent developments in diversity measures allowing to quantify beneficial classifier combinations may also be pursued [73, 74]. Ongoing research into quality of context representation may also provide means to represent TI in a standardized manner.

Another novel mechanism is the ability of our framework to transfer activity recognition capabilities to a new appearing sensor, when this one is not advertising any relevant training for a certain mission. Our approach performs a statistical comparison of the new sensor performance compared to the baseline provided by pre-existing sensors. We have, however, not addressed how to efficiently identify which pairs of sensors are most likely to benefit from the transfer. Future work may include estimating through semantic matching applied to the sensor self-descriptions those candidates and limit the transfer learning to those, which have a high likelihood of benefiting from it. For example, it is likely that a motion sensor deployed on a shoe is able to gather training data for the sensing mission LOCOMOTION, but a humidity sensor mounted in the same position would not be suitable and should be disregarded.

Finally, future work may consider other aspects of opportunistic activity recognition envisioned in [21] but not covered here.

ACKNOWLEDGEMENTS

The project OPPORTUNITY acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number: 225938.

REFERENCES

[1] Puccinelli D, Haenggi M. Wireless sensor networks: applications and challenges of ubiquitous sensing. IEEE Circuit Syst Mag 2005; 5(3): 19-31.

[2] Benini L, Farella E, Guiducci C. Wireless sensor networks: enabling technology for ambient intelligence. Microelectron J 2006; 37(12): 1639-1649.

[3] Verdone R. Wireless sensor and actuator networks: technologies, analysis and design. Academic Press; 2008.

[4] Holmquist LE, Gellersen HW, Kortuem G, *et al.* Building intelligent environments with Smart-Its. IEEE Comput Graph 2004; 24(1): 56-64.

[5] Davies N, Siewiorek DP, Sukthankar R. Special issue: activity-based computing. IEEE Pervasive Comput 2008; 7(2): 20-21.

[6] Weiser M. The computer for the 21st century (reprint). Pervasive Comput 2002; 1(1): 19-25.

[7] Mann S. Humanistic computing: "WearComp" as a new framework and application for intelligent signal processing. Proc IEEE 1998; 86(11): 2123-2151.

[8] Myers B, Hollan J, Cruz I, *et al.* Strategic directions in human-computer interaction. ACM Comput Surveys 1996; 28(4): 794-809.

[9] Kallio S, Kela J, Korpipää P, Mantyjarvi J. User independent gesture interaction for small handheld devices. Int J Pattern Recogn 2006; 20(4): 505-524.

[10] Stiefmeier T, Roggen D, Ogris G, Lukowicz P, Tröster G. Wearable activity tracking in car manufacturing. IEEE Pervasive Comput 2008; 7(2): 42-50.

[11] Kunze K, Wagner F, Kartal E, Morales Kluge E, Lukowicz P. Does context matter? - a quantitative evaluation in a real world

maintenance scenario. In: Tokuda H, Beigl M, Friday A, Brush A, Tobe Y, Eds. Pervasive Computing. Springer Berlin / Heidelberg: Lecture Notes in Computer Science 2009; pp. 372-389.

[12] Starner T, Weaver J, Pentland A. Real-time american sign language recognition using desk and wearable computer based video. IEEE T Pattern Anal 1998; 20(12): 1371-1375.

[13] Harms H, Amft O, Roggen D, Tröster G. Rapid prototyping of smart garments for activity-aware applications. Amb Int Sm Env 2009; 1(2): 87-101.

[14] Tentori M, Favela J. Activity-aware computing for healthcare. IEEE Pervasive Comput 2008; 7(2): 51-57.

[15] Consolvo S, Roessler P, Shelton BE, LaMarca A, Schilit B, Bly S. Technology for care networks of elders. IEEE Pervasive Comput 2004; 3(2): 22-29.

[16] Chaudhury T, Lester J, Borriello G. Assessing wellness by sensing everyday activities and interactions. Workshop Proceedings HCI Challenges in Health Assessment (Conference on Human Factors in Computing Systems) 2005.

[17] Roggen D, Calatroni A, Rossi M, *et al.* Collecting complex activity data sets in highly rich networked sensor environments. Proceedings of 7th International Conference on Networked Sensing Systems 2010; pp. 233-240.

[18] Bao L, Intille SS. Activity recognition from user-annotated acceleration data. Proceedings of the 2nd International Conference on Pervasive Computing 2004; pp. 1-17.

[19] Ward JA, Lukowicz P, Tröster G, Starner T. Activity recognition of assembly tasks using body-worn microphones and accelerometers. IEEE T Pattern Anal 2006; 28(10): 1553-1567.

[20] Conti M, Kumar M. Opportunities in opportunistic computing. Computer 2010; 43(1): 42-50.

[21] Roggen D, Förster K, Calatroni A, *et al.* OPPORTUNITY: towards opportunistic activity and context recognition systems. Proceedings of the 3rd IEEE WoWMoM Workshop on Autononomic and Opportunistic Communications 2009; pp. 1-6.

[22] Tognetti A, Carbonaro N, Zupone G, De Rossi D. Characterization of a novel data glove based on textile integrated sensors. Proceedings of 28th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society (EMBS '06); 2006; pp. 2510-2513.

[23] Calatroni A, Roggen D, Tröster G. A methodology to use unknown new sensors for activity recognition by leveraging sporadic interactions with primitive sensors and behavioral assumptions. Proceedings of the Opportunistic Ubiquitous Systems Workshop (12th ACM Int Conf on Ubiquitous Computing) 2010.

[24] Scholten H, Westenberg R, Schoemaker M. Sensing train integrity. IEEE Sensors Conference 2009; IEEE Computer Society Press 2009; pp. 669-674.

[25] Kumar M. Distributed computing in opportunistic environments. Proceedings of the 6th International Conference on Ubiquitous Intelligence and Computing (UIC '09). Berlin, Heidelberg; Springer-Verlag 2009; pp. 1-1.

[26] Lilien L, Gupta A, Yang Z. Opportunistic networks for emergency applications and their standard implementation framework. Proceedings of the 21st IEEE International Performance, Computing, and Communications Conference 2007; pp. 588-593.

[27] Pelusi L, Passarella A, Conti M. Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. IEEE Commun Mag 2006; 44(11): 134-141.

[28] Lu H, Lane ND, Eisenman SB, Campbell AT. Bubble-sensing: binding sensing tasks to the physical world. Pervasive Mobile Comput 2010; 6(1): 58-71.

[29] Beal J, Bachrach J. Infrastructure for engineered emergence on sensor/actuator networks. IEEE Intell Syst Mag 2006; 21(2): 10-19.

[30] Dressler F. Self-organization in sensor and actor networks. Wiley 2007.

[31] Tsiatsis V, Gluhak A, Bauge T, *et al.* The SENSEI real world Internet architecture. In: Tselentis G, Galis A, Gavras A, *et al.*, Eds. Towards the future Internet - emerging trends from European research. IOS Press 2010.

[32] Kulathumani V, Sridharan M, Ramnath R, Arora A. Weave: An architecture for tailoring urban sensing applications across multiple sensor fabrics. MODUS International Workshop on Mobile Devices and Urban Sensing 2008.

[33] Campbell AT, Eisenman SB, Lane ND, Miluzzo E, Peterson RA. People-centric urban sensing. Proceedings of the 2nd annual international workshop on wireless internet (WICON '06); New York, NY, USA: ACM 2006; pp. 2-5.

[34] Lane ND, Eisenman SB, Musolesi M, Miluzzo E, Campbell AT. Urban sensing systems: opportunistic or participatory? Proceedings of the 9th workshop on Mobile computing systems and applications (HotMobile '08). New York, NY, USA: ACM 2008; pp. 11-16.

[35] Campbell AT, Eisenman SB, Lane ND, Miluzzo E, Peterson RA, Lu H, et al. The rise of people-centric sensing. IEEE Internet Comput 2008; 12(4): 12-21.

[36] Scott J, Crowcroft J, Hui P, Diot C. Haggle: a networking architecture designed around mobile users. Proceedings of the Third Annual Conference on Wireless On-demand Network Systems and Services 2006; pp. 78-86.

[37] Kukkonen J, Lagerspetz E, Nurmi P, Andersson M. BeTelGeuse: A platform for gathering and processing situational data. IEEE Pervasive Comput 2009; 8(2): 49-56.

[38] Fortino G, Guerrieri A, Bellifemine FL, Giannantonio R. SPINE2: Developing BSN applications on heterogeneous sensor nodes. Proceedings of the IEEE Symposium on Industrial Embedded Systems (SIES2009) 2009.

[39] Bannach D, Amft O, Lukowicz P. Rapid prototyping of activity recognition applications. IEEE Pervasive Comput 2008; 7(2): 22-31.

[40] Roggen D, Lombriser C, Rossi M, Tröster G. Titan: an enabling framework for Activity-aware "Pervasive Apps" in opportunistic personal area networks. EURASIP J Wireless Commun Netw 2011.

[41] Dey AK, Abowd GD. The context toolkit: aiding the development of context-aware applications. Workshop on Software Engineering for wearable and pervasive computing 2000; pp. 431-441.

[42] Mamei M, Zambonelli F. Programming pervasive and mobile computing applications with the TOTA middleware. Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications (PerCom) 2004; pp. 263-273.

[43] Eugster PT, Garbinato B, Holzer A. Middleware support for context-aware applications. Middleware for Network Eccentric and Mobile Applications 2009; 305-322.

[44] Gu T, Pung HK, Zhang DQ. A middleware for building context-aware mobile services. Proceedings of the IEEE 59th Vehicular Technology Conference (VTC 2004-Spring) 2004; pp. 2656-2660.

[45] Kjaer KE. A survey of context-aware middleware. Proceedings of the 25th conference on IASTED International Multi-Conference: Software Engineering; Anaheim, CA, USA: ACTA Press 2007; pp. 148-155.

[46] Baldauf M, Dustdar S, Rosenberg F. A survey on context aware systems. Int J Ad Hoc Ubiquitous Comput 2007; 2: 263-277.

[47] Figo D, Diniz PC, Ferreira DR, Cardoso JMP. Preprocessing techniques for context recognition from accelerometer data. Pervasive Mobile Comput 2010; 14(7): 645-662.

[48] Bettini C, Brdiczka O, Henricksen K, Indulska J, Nicklas D, Ranganathan A, Riboni D. A survey of context modelling and reasoning techniques. Pervasive Mobile Comput 2010; 6(2): 161-180.

[49] Lester J, Choudhury T, Borriello G. A practical approach to recognizing physical activities. In: Fishkin K, Schiele B, Nixon P, Quigley A, editors. Pervasive Computing. Springer Berlin / Heidelberg: Lecture Notes in Computer Science 2006; pp. 1-16.

[50] Kunze K, Lukowicz P. Dealing with sensor displacement in motion-based onbody activity recognition systems. Proceedings of the 10th International Conference on Ubiquitous Computing 2008; pp. 20-29.

[51] Förster K, Roggen D, Tröster G. Unsupervised classifier self-calibration through repeated context occurences: is there robustness against sensor displacement to gain? Proceedings of the 13th IEEE Int. Symposium on Wearable Computers (ISWC) 2009; pp. 77-84.

[52] Inc. OGC. OpenGIS sensor model language (SensorML). Implementation Specification 2007.

[53] Indulska J, Sutton P. Location management in pervasive systems. Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003. Australian Computer Society, Inc. 2003; pp. 143-151.

[54] Salber D, Dey AK, Abowd GD. The context toolkit: aiding the development of context-enabled applications. Proceedings of the SIGCHI conference on Human factors in computing systems; ACM 1999; pp. 434-441.

[55] Kurz M, Fercha A. Sensor abstractions for opportunistic activity and context recognition systems. In: Lukowicz P, Kunze K,

kortuem G, Eds. Smart Sensing and Context. Springer Berlin / Heidelberg: Lecture Notes in Computer Science 2010; pp. 135-148.

[56] Kurz M, Hölzl G, Ferscha A, Sagha H, Millán JdR, Chavarriaga R. Dynamic quantification of activity recognition capabilities in opportunistic systems. Proceedings of the Fourth Conference on Context Awareness for Proactive Systems (CAPS2011) 2011; pp. 1-5.

[57] Kurz M, Ferscha A, Calatroni A, Roggen D, Tröster G. Towards a framework for opportunistic activity and context recognition. Proceedings of 12th ACM International Conference on Ubiquitous Computing (Ubicomp 2010), Workshop on Context awareness and information processing in opportunistic ubiquitous systems 2010.

[58] Campbell GA, Turner KJ. Goals and policies for sensor network management. Proceedings of the 2008 Second International Conference on Sensor Technologies and Applications (SENSORCOMM '08). Washington, DC, USA: IEEE Computer Society 2008; pp. 354-359.

[59] Bandara A, Payne T, Roure DD, Gibbins N, Lewis T. A pragmatic approach for the semantic description and matching of pervasive resources. Int J Pervasive Comput Commun 2010; 6(1): 434-446.

[60] Wu X, Jiang T. Matchmaking of goals in intelligent agents based on description logics (DLs). Proceedings of the 2008 International Conference on Intelligent Computation Technology and Automation (ICICTA '08). Washington, DC, USA: IEEE Computer Society 2008; pp. 806-810.

[61] Kuncheva LI, Bezdek JC, Duin RPW. Decision templates for multiple classifier fusion: an experimental comparison. Pattern Recogn 2001; 34(2): 299-314.

[62] Pan SJ, Yang Q. A survey on transfer learning. IEEE T Knowl Data En 2010; 22(10): 1345-1359.

[63] van Kasteren T, Englebienne G, Krose BJA. Transferring knowledge of activity recognition across sensor networks. In: Floréen P, Krüger A, Spasojevic M, Eds. Pervasive Computing. Springer Berlin / Heidelberg: Lecture Notes in Computer Science 2010; pp. 283-300.

[64] Calatroni A, Roggen D, Tröster G. Automatic transfer of activity recognition capabilities between body-worn motion sensors: training newcomers to recognize locomotion. Proceedings of the 8th International Conference on Networked Sensing Systems (INSS); Penghu, Thailand 2011.

[65] Xu L, Krzyzak A, Suen CY. Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE T Syst Man Cyb 1992; 22(3): 418-435.

[66] Minnen D, Westeyn T, Starner T, Ward JA, Lukowicz P. Performance metrics and evaluation issues for continuous activity recognition. Performance Metrics for Intelligent Systems 2006.

[67] Sagha H, Millán JdR, Chavarriaga R. A probabilistic approach to handle missing data for multi-sensory activity recognition. Workshop on Context Awareness and Information Processing in Opportunistic Ubiquitous Systems at 12th ACM International Conference on Ubiquitous Computing 2010.

[68] Sagha H, del R Millán JdR, Chavarriaga R. Detecting anomalies to improve classification performance in opportunistic sensor networks. Seventh IEEE International Workshop on Sensor Networks and Systems for Pervasive Computing (PerSeNS 2011) 2011; pp. 154-159.

[69] Kunze K, Lukowicz P, Junker H, Tröster G. Where am I: recognizing on-body positions of wearable sensors. International Workshop on Location and Context-Awareness (LOCA04); Springer-Verlag 2005; pp. 264-275.

[70] Kunze K, Lukowicz P. Using acceleration signatures from everyday activities for on-body device location. 11th IEEE International Symposium on Wearable Computers 2007; pp. 115-116.

[71] Mizell D. Using gravity to estimate accelerometer orientation. Proceedings of the Seventh IEEE International Symposium on Wearable Computers 2005; pp. 252-253.

[72] Bannach D, Lukowicz P. Integrated tool chain for recording, handling, and utilizing large, multi-modal context data sets for context recognition systems. Proceedings of the 2nd Workshop on Context-Systems Design, Evaluation and Optimisation (CosDEO 2011) 2011; pp. 357-358.

[73] Kuncheva LI, Whitaker CJ. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Mach Learn 2003; 51: 181-207.

[74] Villalonga C, Roggen D, Tröster G. Shaping sensor node ensembles according to their performance prediction within a dynamic planning-based framework. Proceedings of the 8th International Conference on Networked Sensing Systems (INSS); Penghu, Thailand 2011.

Received: February 19, 2011 Revised: March 25, 2011 Accepted: April 01, 2011