A Tool-Based Methodology For Long-Term Activity Monitoring

Rafik Belloum, Charles Consel, Nic Volanschi

To cite this version:

Rafik Belloum, Charles Consel, Nic Volanschi. A Tool-Based Methodology For Long-Term Activity Monitoring. PETRA'20 - Pervasive Technologies Related to Assistive Environments, Jun 2020, Corfu Island, Greece. 10.1145/3389189.3392609 . hal-02551364

HAL Id: hal-02551364

https://hal.archives-ouvertes.fr/hal-02551364

Submitted on 30 Apr 2020

A Tool-Based Methodology For Long-Term Activity Monitoring

Rafik Belloum
Inria Bordeaux, France
rafik.belloum@inria.fr

Charles Consel
Bordeaux INP, France
charles.consel@bordeaux-inp.fr

Nic Volanschi
Inria Bordeaux, France
eugene.volanschi@inria.fr

ABSTRACT

In recent years, remarkable progress has been reported in the field of activity monitoring. However, despite significant breakthroughs in recognizing activities from sensor data, there is still a great deal more to accomplish, especially compared to fields pursuing related goals, such as computer vision and speech recognition. A key factor to move activity monitoring forward, is to enable researchers to build on each other's work more systematically via reproducible research. Besides providing sensor data, reproducibility in activity monitoring requires all aspects of a result to be available to the research community, including collection, processing and interpretation of measurements.

This paper presents a tool-based methodology, dedicated to monitor the activities of daily living of older adults, that supports reproducible research. This methodology covers the key steps to defining a monitoring process of these activities, from sensor measurements to actionable activity information. These steps are uniformly described with concise and high-level rules. Additionally, to allow caregivers to monitor older adults' functional decline and to determine what assisting support is needed, our methodology includes a visualization tool, dedicated to handling user activities longitudinally.

The proposed approach is validated by a set of rules dedicated to monitor activities of community-dwelling, older adults in their sensor-equipped homes. A preliminary study[1] has been conducted to evaluate the intra- and inter-participant consistency of the results produced by our methodology, using longitudinal datasets, collected over several months. Using Signal Detection Theory, it has shown that our monitoring rules mostly produced the same interpretations as an expert in activity analysis, who manually analyzed the sensor datasets.

CCS CONCEPTS

• **Software and its engineering** → *Domain specific languages*; **Rapid application development**; • **Human-centered computing** → Ubiquitous computing; *Visualization techniques*; • **Social and professional topics** → **Assistive technologies**.

KEYWORDS

Older adults, smart home, assistive computing, activity recognition, sensor failure detection, visualization, reproducibility

ACM Reference Format:
Rafik Belloum, Charles Consel, and Nic Volanschi. 2020. A Tool-Based Methodology For Long-Term Activity Monitoring. In *The 13th PErvasive Technologies Related to Assistive Environments Conference (PETRA '20), June 30-July 3, 2020, Corfu, Greece.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3389189.3392609

1 INTRODUCTION

In recent years, a host of pervasive computing technologies have been developed and deployed, sensing a range of user activities. Such technologies as connected watches, multi-purpose ambient sensors, and conversational agents offer remarkable opportunities for context-aware services to enhance the functional independence of older adults and reduce the caregiver burden [2, 5]. Context awareness revolves around monitoring user activities. For older adults, this monitoring allows to detect whether activities of daily living (ADLs) are completed (*e.g.,* dressing, bathing, sleeping, and cooking); this information serves as a reliable indicator of the level of functional independence of an individual [16].

Over the years, many research approaches have attempted to monitor ADLs. Increasingly, research approaches have either been based on sensors or vision. The latter approach faces great challenges to go beyond laboratory settings. Indeed, a camera is largely regarded as unsuitable and unacceptable in real-life due to privacy concerns and the computational cost of video processing [13]. During the past few decades, steady progress has been made in developing sensor-based approaches, capable of recognizing a range of activities. Although promising, this research is still in its early stage and needs to address some key challenges.

Monitoring activities in the wild for a long period of time. Monitoring activities in a real home often defeats conceptual models developed in a controlled environment. Specifically, sensors do fail, especially when they are deployed over a long period of time. Furthermore, detecting activities requires to handle a range of unexpected situations, which only occur when the home occupant is continuously monitored, across several months. A realistic setting calls for an *iterative process* to refine the analysis of sensor data, driven by a synoptic view of user activities.

Comprehensible and accessible data processing algorithms. A prerequisite to reproducibility is that data processing algorithms be accessible to and comprehensible for other researchers. Unfortunately, processing event-based contexts is often poorly addressed by mainstream/general-purpose programming languages (GPLs), or by domain-specific languages (DSLs), such as CEP (Complex Event

[1] The present case study is publicly available at the following URL: https://gitlab.inria.fr/rbelloum/reproducibilitymonitoring.git

PETRA '20, June 30-July 3, 2020, Corfu, Greece

ACM ISBN 978-1-4503-7773-7/20/06…$15.00
https://doi.org/10.1145/3389189.3392609

Processing) [23]. Consequently, resulting algorithms written in existing languages of both types tend to lack both conciseness [24] and comprehensibility, making it difficult for researchers to evolve their data processing algorithms and to build on each other's work. These algorithms should be written in a *dedicated language*, which addresses these shortcomings.

Scaling up the monitoring of activities. Developing a system that monitors the daily activities of a user in the wild is very challenging. A methodology claiming to support the development of such systems should be applied to a realistic case study and evaluated with respect to 1) its ability to overcome sensor failures, 2) its support to aid the researchers cover unexpected user-activity patterns, and 3) its effectiveness in making sizeable sensor data actionable.

Our approach

This paper introduces a tool-based methodology that covers the key aspects of an activity monitoring system.

An iterative process to define the analysis of sensor data. To compensate for sensor failures and reliably detect activities, an iterative process is introduced; it supports a stepwise refinement of the analysis of the sensor data. This process consists of applying analysis rules to realistic sensor data and checking their output against typical user-activity patterns. To support this process, a visualization tool is used by the rule developer to ensure that the detected activities have an overall consistency. In practice, this process allows to gradually introduce knowledge about user-activity patterns in the analysis rules. Note that the visualization tool is also used by caregivers to monitor older adults longitudinally.

Using a dedicated language. To allow analysis rules to evolve during the iterative process, our approach revolves around a domain-specific language, called Allen [24], dedicated to defining rules that analyze sensor data. Specifically, this DSL is used to write rules that detect sensor failures and activities of daily living. Because of the dedicated nature of this DSL, analysis rules are concise and high-level, facilitating their evolution. As a byproduct, the use of this DSL makes the rules more comprehensible to other researchers, contributing to research reproducibility.

Putting the methodology into practice. Our tool-based approach has been applied to realistic, sensor data from real homes of five older adults, collected over several months. These rules to detect sensor failures and specific activities have been refined using the proposed iterative process. The results have been validated, using signal detection theory, by comparing the results of our rules with a human observer. These rules mostly produced the same interpretation as the human judge, who manually analyzed the datasets.

Overview

The rest of the paper is organized as follows. Section 2 reviews previous works on sensor-based activity monitoring. Section 3 briefly presents the field experiment from which our sensor data were extracted. Section 4 describes the proposed approach for reproducible and longitudinal activity monitoring. Section 5 describes a case study to allow the reproducibility of concise rules for activity monitoring. Section 6 demonstrates the accuracy of our knowledge-based rules from a natural-setting data sensor. Section 7 concludes and discusses future work.

2 RELATED WORK

Firstly, the characteristics of home-based activities are investigated in the context of our target population, namely older adults, and the impact of aging on these activities. Secondly, previous studies in the area of monitoring activities are reviewed, examining what types of sensors and experimental settings were used. Finally, the barriers to reproducibility in research on activity monitoring are discussed.

2.1 Older-adult daily activities

The autonomous performance of ADLs is an important factor to promote independence in everyday activities [7]. To monitor and assess ADLs, they need to be characterized. To do so, a number of dimensions can be used, including the location where they take place, the time of day at which they occur, and the environment interactions they entail [12]. For example, *sleeping* takes place in the bedroom, *dinner* occurs in the evening, *etc.* For older adults, ADLs are increasingly routinized with age decline, compensating for decreasing cognitive resources [1]. Caroux *et al.* were the first to leverage this situation and to develop a knowledge-based approach to verifying whether activities of interest are performed [2]. Specifically, Caroux *et al.* use the declarations by older adults on their daily routines to determine how to verify activities of interest (*e.g.,* types of sensors, number of sensors, sensor locations), instead of inferring them.

2.2 Range of sensors

There is a large variety of sensors that can be used to monitor activities in a home. Sensors are typically split into two categories: ambient sensors, which instrument the environment, and wearable sensors, which instrument the user. Ambient sensors can either be wall-mounted (*e.g.,* motion detection sensors) or placed on objects (*e.g.,* contact sensors placed on doors and cupboards). Wearable sensors can be a bracelet detecting falls or an RFID tag tracking the location of a user. Wearable sensors are often said to be unsuited for older adults, who may not accept them because of their intrusive nature. In contrast, except for webcams, ambient sensors can blend into the environment and sustainably contribute to detect activities [15].

2.3 Experimental settings

Most available sensor data targeting activity recognition are recorded in a controlled environment dedicated to experimental studies [15, 19]. In such settings, multi-day experiments are typically conducted with students, who live in the controlled environment for a few days, possibly performing pre-defined tasks. If older adults are recruited, they usually participate to studies which only last for a few hours. Because such an environment is unfamiliar to them, their performance in executing activities is unlikely to match their performance at home, where they have developed strategies to compensate for decreasing cognitive resources [2].

2.4 Sensor data processing

The research in processing sensor data to monitor activities can be decomposed into three topics: 1) the activity recognition techniques, 2) the detection of sensor failures, as well as user routine deviations, and 3) the programming support for processing sensor readings.

2.4.1 Activity recognition. Approaches to activity recognition are mainly based on machine learning or driven by user knowledge [8].

Machine-learning approaches use statistical and probabilistic methods to learn activity models from datasets collected by ambient sensors, which monitor environment interactions. The approaches are becoming mainstream in the domain of activity recognition. One particular advantage is that they allow the modelling of uncertainty and the handling of temporal information. However, sensor data used for machine learning approaches usually need to be collected at a large scale to be statistically robust. As well, such approaches rely on an accurate and labor-intensive process to label activities and evaluate the performance of recognition models [15]. The cost of the labelling task and its sensitivity to changes over time and across individuals, which occur in real homes [15], may explain why machine-learning approaches are primarily explored in controlled environments during short experiments (*i.e.,* a few weeks), as illustrated by Dawadi *et al.*'s work [8, 9].

As introduced earlier, a knowledge-driven approach relies on routine declarations of users in their home to create activity models, which can be checked with a minimal set of sensors, placed at strategic locations [2]. This approach was developed in the context of the HomeAssist project (see Section 3) and has proven to be effective in naturalistic environments (*i.e.,* real homes), across a sizeable group of older adults (*i.e.,* 140 participants), and over a long period of time (*i.e.,* 12 months) [3]. Noticeably, it has only been applied to single-occupant homes.

Finally, knowledge-based rule and probabilistic inference have been combined in hybrid approaches such as Computational State Space Models, and more recently Computational Causal Behaviour Models [25]. On the one hand, by virtue of the knowledge-based rules component, such approaches may achieve more robustness to unseen cases than a pure machine learning approach. On the other hand, due to the probabilistic component, they achieve robustness to sensor noise. However, the probabilistic part of the model still requires training data to be used; they have to be recorded and manually annotated.

2.4.2 Anomaly detection. In a real-life setting and over a long period of time, some of the sensors installed in a home do experience failures and malfunctions, which may result in misleading interpretations when activities are being monitored (*e.g.,* a lost sensor packet signalling a door closed). Machine learning approaches rely on sensor data to construct activity models. Therefore, they are sensitive to sensor failures and malfunctions, which can negatively interfere with the training process. There are publicly available datasets from experimental studies in ambient assisted living (Kasteren [22], Casas [4], Placelab [15]). Although these datasets include labelled activities for activity detection purposes, none of them include any labeling of data produced by faulty sensors [11]. As a result, research on activity monitoring in the presence of sensor failures and malfunctions have required researchers to manually inject such events *a posteriori* in existing datasets. Although this approach is a step toward more realistic datasets, it remains a simulation, which may not be representative of the extended range of sensor anomalies, occurring in a real home, over a long period of time [11].

To construct their activity model, knowledge-driven methods do not rely on data but only use information about the activities. Existing systems using these types of methods do not include anomaly detection techniques because their algorithms for activity inference are designed to be directly executed on the datasets, as reported in the literature [12, 17, 18]. Yet, it has been shown that not only must activity monitoring detect abnormal sensor events, due to anomalies, and discard them, but it must also recognize abnormal user behaviors, such as sudden changes in the routines of an older adult due to health issues. Such situations are paramount to AAL and have been studied by Tran *et al.* [21], who have defined four types of abnormal behaviors:

- Known behavior in a deviating spatial context (*e.g.,* sleeping in the living room)
- Known behavior occurring at a deviating moment in time (*e.g.,* leaving home at abnormal time, having dinner unusually late)
- Known behavior with an abnormal duration or occurrence (*e.g.,* sleeping until noon, or going to the toilet twice as many times as before)
- Behavior resulting in abnormal/unexpected sensor firing patterns (*e.g.,* a fall resulting in an extended period of mute sensors).

These types of abnormal behaviors further demonstrate the key role of knowledge about user routines to make the distinction between sensor anomalies and abnormal behaviors.

2.4.3 Programming support for data processing. Most existing approaches to developing sensor-based, context-aware services use GPLs. These languages do not provide specific support for encoding activity-detection logic in terms of sensor firing patterns. This difficulty is exacerbated by the need to customize the activity-detection logic with respect to the older adult's routines, home setting, and lifestyle. For example, everyday at noon Bob gets ready to have lunch; he opens the fridge to get one of his daily-delivered meals and starts the microwave to warm it up. In contrast, earlier in the morning, Alice opens the cupboards and the fridge to take out ingredients and cook herself a meal using the stove. As illustrated by Bob and Alice, activity detection requires (1) to encode activity detection logic with respect to sensors and event conditions, and (2) to take into account inter-individual variations thereof, which requires developing many variations of such logic for each activity. This approach often results in making the code complex and tedious to develop and evolve. This issue even concerns DSLs for complex event processing (CEP), whose syntax and semantics can quickly obfuscate the detection logic [23].

An alternative is offered by IFTT (If This Then That) and a variant such as AppsGate. They are prime examples of end-user DSLs [6], allowing non-programmers to easily express services, which combine a range of sensors and actuators, at the expense of various restrictions. For example, conditions only refer to one event and a single state. Although convenient for simple scenarios (*e.g.,* home automation), such DSLs are too limited for AAL scenarios.

Furthermore, as observed by Huang *et al.*, specifying services in IFTT is difficult because the notion of event and state are frequently confused by users [14].

To sum up, most approaches to programming activity-detection logic, whether using a GPL or a DSL, do not scale with the variations of user specificities. This shortcoming hampers the comprehensibility of the resulting code, which, in turn, becomes an obstacle toward making research on activity detection reproducible. To pursue reproducibility, our approach relies on Allen, a domain-specific language for programming context-aware services [24]. It has been demonstrated that Allen provides a promising solution to the above mentioned issues, compared to other approaches: it allows to express context detection logic at a higher level and much more concisely, when programming the same services. Also, compared to complex event processing DSLs, it allows less restricted compositions and supports reusable abstractions, which facilitates expressing complex logic.

2.5 Summary

A brief tour of activity monitoring of older adults in their home has been given. The challenges that impede the reproducibility of this research have been examined, namely, (1) the comprehensibility of algorithms for processing sensor data; (2) the lack of extended studies; and (3) the lack of publicly available datasets. These challenges have prompted us to introduce a tool-based methodology to conduct reproducible research in activiy monitoring.

3 BACKGROUND

Before introducing our methodology, we present the experiment from which our datasets were extracted. This experiment was part of the HomeAssist project [3], aimed to support aging in place by deploying an assistive computing platform [?] in the home of older adults living alone. This platform relies on a range of devices (*e.g.,* sensors and actuators) and runs a set of services, which monitors and assists seniors in their everyday activities.

To prescribe these services a number of stakeholders are involved; they include the older adults and their family, their professional caregivers, and more broadly, occupational therapists, experts in aging, and psychologists. These stakeholders gather the specific needs of the older adult to identify which activities require to be monitored and possibly assisted. Each activity of interest is then sketched by the older adult in their home to determine what interactions with the environment need to be measured to recognize it reliably. In this project, three kinds of applications were provided: (1) applications to monitor ADLs and provide assistance when they are not performed (*e.g.,* reminders, task prompting, *etc.*); (2) alert notifications when security issues are detected (*e.g.,* the entrance door left open) and (3) applications to support social interactions (*e.g.,* collaborative games, video conference, *etc.*). These services use a minimal set of wireless sensors, placed at strategic locations in the home to detect a pattern of interactions indicative of an activity of interest. The sensors typically signal a presence in an area (*i.e.,* motion detectors), detect a door being operated (*i.e.,* contact sensors), and an electric-powered device being used (*i.e.,* electric meters). The first two kinds of sensors are battery powered to prevent any constraints in placing them. The number and location of sensors vary across home layouts, user needs, and user routines. HomeAssist was used in two field studies and was deployed in 140 homes of older adults, aged 80 years and older, living alone, during a maximum period of 24 months. These studies showed a positive impact of HomeAssist on participants in terms of daily autonomy, self-regulation and empowerment [10].

Table 1: HomeAssist sensors and their functions

Room	Sensor ID	Function
Kitchen	EMeter_Coffeemaker	Coffee maker in use
	EMeter_Microwave	Microwave in use
	ContactS_Cupboard	Cabinet door open
	ContactS_Fridge	Fridge door open
	MotionD_K	Presence
Entrance	ContactS_E	Door open
	MotionD_E	Presence
Bedroom	EMeter_L	Bedside lamp in use
	MotionD_B	Presence
Bathroom	MotionD_Ba	bathtub presence
	MotionD_S	Shower presence
Toilet	MotionD_T	Presence
Living room	MotionD_L	Presence

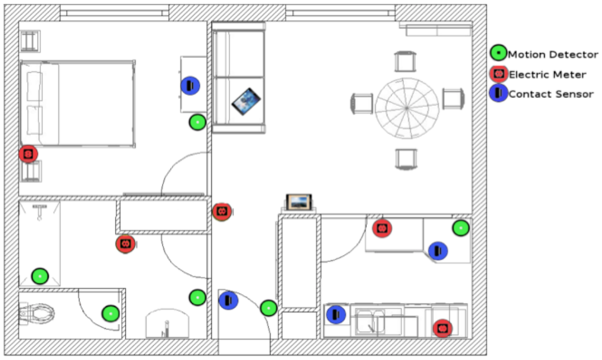

Figure 1: Example of an apartment layout with sensors

Because our methodology is illustrated by datasets from the HomeAssist project, let us now introduce the name and function of the sensors used in this platform, as well as the rooms of interest. This information is presented in Table 1, where each room of interest is listed in the first column. Each room is equipped with sensors, whose names are defined in the second column (Sensor ID); we later use these names to define monitoring rules. Each sensor is associated with a function, which defines the meaning of the interaction it measures. Figure 1 illustrates an example of an apartment layout equipped with sensors.

4 METHODOLOGY

This section presents the proposed tool-based approach, which (1) processes longitudinal sensor data with respect to monitoring rules, aimed to detect activities and sensor failures, and (2) provides a tool capable of visualizing activities over a long period of time for screening purposes.

Table 2: Output of monitoring rules detecting activities and sensor failures

Start date	End date	Label
2017-08-01 08:09:02	2017-08-01 08:50:37	outing
2017-10-20 04:56:34	2017-10-20 05:00:20	toilet
2017-11-09 21:15:31	2017-11-13 14:21:22	toilet_failure
2017-12-27 21:00:00	2017-12-28 06:32:48	sleep_quiet
2018-02-07 14:18:25	2018-02-10 02:59:27	platform_failure
2018-08-01 15:05:56	2018-08-06 08:42:20	bed_failure
2018-08-03 13:56:45	2018-08-12 20:03:51	door_failure

Let us first examine the steps required to define monitoring rules, according to the different dimensions of the activity monitoring system: activity detection and sensor failure detection.

4.1 Defining monitoring rules

Defining monitoring rules is an iterative process, which consists of four steps. Firstly, the developer writes a rule in a dedicated language (see below) to detect an activity or sensor failures by processing binary sensor data. As illustrated in Table 2, each rule produces a label, denoting an activity or sensor failures, for a given period during which a situation has been detected. Note that the detected situations may overlap in time, as illustrated by the last two lines of Table 2, labelled 'bed_failure' and 'door_failure'. Thus, each rule is executed independently by processing its input sensors.

After writing the first version of the rule, the developer carries out a feasibility study. Specifically, the rule undergoes preliminary testing by applying it to several sets of real sensor data and matching a sample of its results against a manual analysis of the corresponding sensor data. In doing so, this phase determines whether the detection of a situation of interest can be formulated in a rule, which combines one or more sensors at strategic locations; and, whether a rule has the potential of producing reliable information. If this feasibility study is successful, then an iterative process to refine the rule is initiated; it identifies the patterns of sensor data that can lead to erroneous labels, and refines the rule to cover the various homes and user specificities. The completeness of these patterns depends on the rigor used to conduct the iterative process and the representativeness of the data. As a final step, the accuracy of a rule is evaluated against a human observer, to ensure that it produces the same interpretation as a human observer.

Let us now illustrate the first three steps of the proposed approach by defining a rule that detects visits to the toilets. The fourth step is examined in Section 6.

Writing a rule. The goal is to define a rule that detects the toilet activity by measuring the user's presence via a motion detector, placed inside the toilet. Recall that, because the HomeAssist project is leveraged, this work focuses on single-occupant dwellings. In its simplest form, the rule for detecting toilet visits can be written in Allen as follows.

```
toilet:
  "MotionD_T"
```

The rule is named `toilet`, is introduced with a colon (':'), and produces Label "toilet" whenever the condition of the rule is true; that is, when a motion is detected in the toilet via the motion sensor named `MotionD_T`. Generally speaking, when a sensor occurs in an Allen rule, its execution produces all the time intervals during which its value is true (*i.e.,* its value is 1).

Feasibility study. Once a first version of a rule is defined, the developer needs to apply it to sensor data across several participants and manually analyze the results to gather and generalize special cases that may have occurred. In our example, a typical situation that needs to be handled is the loss of the sensor event indicating that the user left the toilet. Loss of sensor data must be addressed when measurements are performed in a natural setting; it is typically caused by a low battery condition, temporary loss of radio transmission (or radio reception on the sensor gateway side), and packet collisions. In our example, if not properly handled, the loss of this information means that the visit to the toilet is endless. Another situation observed on ecological sensor data is the occurence of many, very short toilet visits; that is, visits separated by less than a minute. This situation is caused by a user who is motionless during enough time so as to cause the sensor to indicate that the room is unoccupied, until a new motion is detected shortly afterwards.

These two situations have the potential to cause the first version of our rule to produce erroneous information. However, they do not compromise the feasibility of our rule to detect toilet visits because they can be compensated by introducing simple conditions. This refinement is conducted next.

Iterative refinement. The loss of the sensor event indicating the exit from the toilet can be addressed by setting an upper limit on the duration of a toilet visit. This limit allows our rule to compensate for sensor faults and transmission losses and to reset its state so as to detect future toilet visits. Specifically, a toilet occupancy is considered valid, if it does not last more than (Operator '<=') a given duration (Parameter `T1`). This parameter is set to the appropriate value (*e.g.*, 20 min) depending on the user specificities, which can be determined by examining the sensor data. When toilet visits are longer than the duration limit, they are discarded by the rule. The new version of our rule is defined below.

```
toilet:
  "MotionD_T"<= T1
```

To circumvent the second situation (*i.e.,* close, short visits due to a lack of motion), we need to group together intervals of motion separated by short pauses. First, let us define a rule that recognizes a short pause, as shown below.

```
toilet_pause:
   holds(~any_motion_up, ~"MotionD_T" <= T2)
```

Rule `toilet_pause` is true when there is no motion in the toilet during less than `T2` minutes and no motion is detected anywhere else in the home. Absence of motion is expressed using the negation operator (˜). Short absences of motion are filtered by Operator '<='. Further filtering is performed by Operator `holds(p,q)`, which gathers the time intervals during which `q` is true and only keeps the ones for which `p` holds. In our example, Operator `holds` allows to select short absences in the toilet (`q`) during which no movement is detected elsewhere (`p`). This last condition (`p`) is defined by (the negation of) Rule `any_motion_up`, which selects periods during which no motion is detected anywhere in the home (its definition is given in Section 5). The relationship between the filtering operators of the DSL and the original Allen logic is detailed elsewhere [24].

Because we assume single-occupant dwellings, a presence detected in the toilet followed by no motion in the home indicates that the user has not left that room. Extending Rule `toilet` with Rule `toilet_pause` gives the following definition, using the logic 'or' operator (|).

```
toilet:
  "MotionD_T" <= T1 | holds(~any_motion_up, ~"MotionD_T"
     <= T2)
```

This iterative refinement of Rule `toilet` was completed when sensor data of representative participants were successfully labelled.

4.1.1 *Sensor-failure detection.*

To address sensor failures, dedicated rules need to be defined. Their aim is to report periods during which sensor failures occurred, as shown in Table 2. The definition of such rules follows the same steps as the ones to detect activities, namely, writing a rule, a feasibility study, and an iterative refinement.

For sensors providing a failure detection mechanism, such as a heartbeat, detecting failures is straightfoward, as shown in the following rule.

```
toilet_failure:
   "MotionD_T.CommFailure"
```

This expression annotates the periods during which the sensor is out of service, as exposed by Attribute `CommFailure` of Sensor `MotionD_T`. This attribute is available on any sensor but needs to be refined because we found that in many cases it is not reliable. An alternative is to define a faulty sensor as one that does not emit information for an extended period of time, which depends on the location of the sensor and the environment interaction it is measuring. For example, the toilet is typically visited many times everyday. Consequently, we can introduce a rule to detect the failure of the motion detector of the toilet as follows.

```
toilet_failure:
 ~"MotionD_T" >= T | "MotionD_T" >= T
```

This rule states that the toilet sensor fails if it is inactive for more than time `T` (first term) or active for more than time `T` (second term). Parameter `T` is typically set to 1 day or more. Note that a motion detector is (in-)active during an extended period of time if the message indicating a lack/presence of motion was lost or the sensor is locked in a given state and needs to be reset.

Further applying this version of `toilet_failure` to sensor data and manually analyzing the results reveal another issue: some periods do not show any toilet visits because of outings of the home occupant that last more than time `T` (*e.g.*, one or more days). To account for this situation, a rule detecting outings (defined in Section 5) needs to be included. The new version of Rule `toilet_failure` is defined below, using the logical 'and' operator (&) to skip outings.

```
toilet_failure:
 (~"MotionD_T" & ~outing) >= T |
 ("MotionD_T" & ~outing) >= T
```

This version of the rule was applied to a variety of sensor data and produced correct results when manually checked.

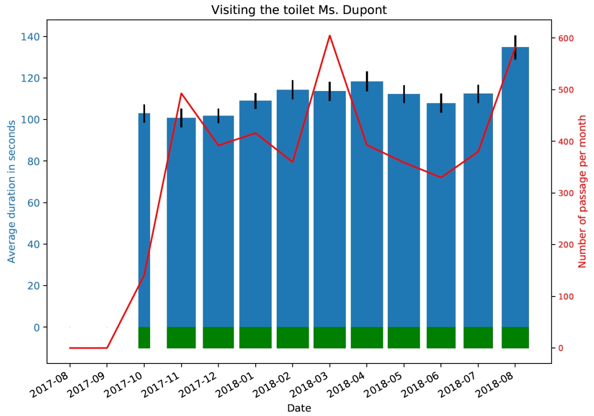

Figure 2: Visualization of toilet activity

4.2 Long-term visualization

Because of the study duration of HomeAssist, applying monitoring rules to sensor data produces massive amounts of activity information. This situation raises the need to provide a synoptic view of these activity information to allow caregivers to analyze them, identify trends or events of interest, and take action if necessary. We propose an approach to visually characterizing a user activity during a given period of time. An example is shown in Figure 2. Specifically, the blue bars represent the average duration of the activity, and the red line indicates its frequency (*e.g.,* number of visits per month). However, the detected periods of sensor failures obviously impact the significance of the average duration of the activity. To visually account for this aspect, the width of the bars is adjusted with respect to the extent to which sensor failures occurred: the fewer the failures, the thicker the bar, indicating a more significant average value of activity duration.

However, when no blue bar is displayed in the graph for a given month, this may correspond to two different cases: *Situation 1* – no activity is performed at all during the month, although the sensor worked properly during this period, or *Situation 2* – the sensor failures cover the whole period, hence there is no information about the activity. To avoid the ambiguity between these situations, we added green witness bars at the bottom of the graph, independently of the presence of the blue bars, to separately indicate the periods when the sensors were working, via their width, as for the blue bars. Thus, Situation 1 results in a green bar without a blue bar (not occurring in Figure 2); and Situation 2 results in the absence of both bars (first two months in the figure). Because erroneous data are filtered out, they do no impact the computed average values.

Example. Figure 2 represents the toilet activity of Ms. Dupont. The x-axis represents the period at which the sensor data were collected. The y-axis on the right side, represents the number of toilet visits per month (the red line). The y-axis on the left side, represents the average duration of activity for this participant (the blue bars), which varies between 100s and 140s. Thanks to the green witness bars and their width, one can see the periods during which the toilet detector properly worked in this user's home. During August and September 2017, there is no activity information: no green (hence no blue bars) shown in the graph. This unambiguously indicates that during these two months, this activity could not be measured. The caregiver cannot misinterpret this period as a routine deviation: information about the activity is simply missing. The vertical lines around the top of each bar represent the standard deviation of the averaged values. Considering the small values of the standard deviation in relation to variations between bars, the results of our rule can be considered as significant.

5 CASE STUDY

This section applies the proposed methodology to a realistic case study. Specifically, two activities of daily living are examined, namely, outings and sleeping. This presentation is used to show how our approach contributes to improving replicability and reproducibility via the definition of concise, high-level monitoring rules, which can be made available to other researchers, as well as the sensor data. In fact, the present case study is publicly available at the following URL: https://gitlab.inria.fr/rbelloum/reproducibilitymonitoring.git This reproducibility kit contains (1) the complete dataset of one participant, covering the whole year 2017 with 95,157 sensor measurements, and (2) instructions for replaying the activity detection and visualization phases explained in this paper. The dataset of the other participants could not be disclosed for privacy reasons.

5.1 Outings

Let us first define the rules for each of the monitoring activities addressed by our case study. To do so, the notion of user-defined operator provided by Allen is introduced; it allows some structuring and reuse in programming monitoring rules. User-defined operators can be seen as function definitions in mainstream programming languages. They allow rules to be reused and parameterized, allowing them to be customized with respect to user specificities. A user-defined operator is introduced with Construct `def` followed by a name and optional parameters, delimited by square brackets.

5.1.1 Activity detection. We begin by defining the user-defined operator `outing_period` to delimit the period during which no activity occurs in a home between two consecutive door openings.

```
def outing_period =
  holds(no_activity,
    between(dn("ContactS_E"), up("ContactS_E")))
```

More precisely, an outing period begins when the entrance door is closed (*i.e.,* user’s departure), and ends when it is opened again (*i.e.,* user’s return). Technically, Operators `up` and `dn` select the time when a sensor goes from 0 to 1, respectively from 1 to 0. Operator `between(p,q)` selects any time interval between a period when p is true and the subsequent period when q is true. Therefore, the expression means that, if after Door sensor `"ContactS_E"` has produced Value 0 (state ‘closed’), there is no other activation of any sensor inside the home until the next activation of `"ContactS_E"` (state ‘open’), an outing has occurred. The term `no_activity` is another user-defined operator, detailed later. It is true whenever no sensor is activated in the home.

Following the proposed methodology, samples of sensor data of participants are manually analyzed. It was noticed that our rule detected some short outings, which probably correspond to the user taking out the trash or picking up the mail. After consulting with our experts in aging, these short absences were not deemed proper outings because they likely did not involve much physical activity nor social interaction. To skip such absences, the operator definition was extended with a minimal time of absence `T` before declaring an outing. This extension took the form of a parameter to the user-defined operator definition, as shown below.

```
def outing_period[T] =
  holds(no_activity,
    between(dn("ContactS_E"), up("ContactS_E")) > T)
```

Parameterization thus allows this rule to be customized with respect to the elapsed time before a departure is considered an outing. Such customization is shown below with the definition of Rule `outing`.

```
outing:
  outing_period[10min]
```

Thanks to the domain-specific nature of Allen, it offers a built-in type and related constants to express time (*e.g.,* `10min`, `21hr`).

The user-defined operator `no_activity` is defined below, following some auxiliary definitions. A period is labelled as `no_activity` when no electric appliances, contact or motion sensors are activated in a home. This situation is defined by the following set of rules.

```
def any_emeter_up =
  any_up("EMeter_Microwave", "EMeter_Coffeemaker", "
     EMeter_L")
def any_contact_sw =
  any_sw("ContactS_B", "ContactS_Cupboard", "ContactS_E",
      "ContactS_Fridge")
def any_motion_up =
  any_up("MotionD_B", "MotionD_Ba", "MotionD_E", "
     MotionD_K", "MotionD_L", "MotionD_S")
def any_activity =
  any_motion_up() | any_emeter_up() | any_contact_sw()
def no_activity =
  ~any_activity()
```

Operators `any_up` and `any_dn` are the n-ary version of Operators `up` and `dn` introduced earlier. A third operator called `sw`, and its n-ary version `any_sw` signal the moments when a sensor switches value (up/down).

5.1.2 Sensor-failure detection. We now investigate the sensor failures that can compromise the detection of outings, as was done for toilet visits. Here, the key component is the contact sensor monitoring the entrance door. Because we follow the same logic as the final rule for toilet failures, we introduce a user-defined operator that encapsulates these failures, and parameterize it with 1 week (*i.e.,* 168 hours).

```
def sensor_failure[T](s) =
  (~s & ~outing) >=  T | (s & ~outing) >= T
door_failure:
  sensor_failure[168hr]("ContactS_E")
```

As can be noticed, user-defined operators can not only be parameterized with time constants, but can also take sensor name parameters. To visually distinguish these two types, time parameters are given between square brackets, while sensor name parameters are given between parentheses.

This rule determines that the contact sensor has failed, if the user is home (*i.e.,* `~outing`) and the contact sensor of the entrance door has not been activated during a given time `T`.

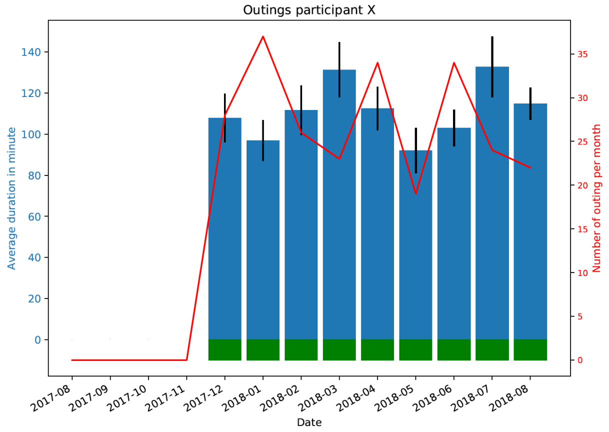

Figure 3: Visualization of outings

5.1.3 Visualization. Figure 3 visualizes the information produced by our rules for outings and failures of the entrance door sensor. The y-axis on the right side represents the number of outings per month (the red line), and the y-axis on the left side represents the average duration of this activity (the blue bars). As can be noticed, during the first 4 months of the monitoring, the displayed graph does not show any outing. However, the absence of a green bar during these months indicates that the contact sensor of the entrance door did not work during this period. This sensor functioned properly for the remainder of the year, as shown by the width of the (blue and) green bars. We also notice that our rules produced consistent measurements of the outing activity: the participant made between 22 and 37 outings per month, lasting from 1h30 to 2 hours. In our experience, this consistency is a key factor in giving confidence to monitoring rules.

5.2 Sleeping

It would be unrealistic to aim at detecting actual sleep of an individual solely with ambient sensors. Instead, the goal is to detect when the user spends some quiet time during the night in their bedroom. This activity should give an indication of how much sleep an individual is getting.

5.2.1 Activity detection. Let us incrementally define our notion of sleep. First, this activity is assumed to occur in the bedroom, and thus, any motion detected in another room may contradict a sleep activity. This situation is covered by the following fragment of rule introduced below.

```
let any_motion_up_but_bed =
  any_up("MotionD_E", "MotionD_K", "MotionD_L",
         "MotionD_S", "MotionD_T")
```

Note the use of a local variable, introduced by Construct `let`. Local variables are used to factorize some logic without parameters. Local variable `any_motion_up_but_bed` is then included in another rule fragment to define a segment of continuous presence in the bedroom, in much the same way as toilet visits (Rule `toilet` in Section 4.1).

```
let sleep_segment =
  holds(~any_motion_up_but_bed,
        ("MotionD_B" >= 30min) <= 10hr |
        (~"MotionD_B" >= 30min) <= 10hr)
```

Specifically, Rule `sleep_segment` produces a segment of sleep, if motion has not occurred anywhere else but in the bedroom, and if user presence in the bedroom is sensed, using the motion detector, during at least 30 minutes but no more than 10 hours. Alternatively, because of the nature of the target activity, a segment of sleep is also produced if there is an absence of motion in the bedroom that lasts between 30 minutes and 10 hours. These two alternatives take into account intra- and inter-individual variations of motion patterns during sleep. Importantly, embedding the disjunction of the two patterns in Operator `holds` (unlike the rule for toilet visits) is the result of the iterative refinement in our methodology, which revealed and fixed inconsistent results.

As our approach is knowledge driven, it leverages the personal routines of the user and thus gathers their usual sleep time slots. This piece of knowledge is introduced as a variable, named `night`. For example, a participant declares that he usually goes to sleep at 9:00PM and wakes up at 8:00AM. Here is the definition of the `night` time slot for this user as a variable in Allen:.

```
let night = slot_2017[21hr, 8hr] | slot_2018[21hr, 8hr]
```

The above definition uses Allen operators `slot_YYYY` generating periodic signals, which are true between the given times of the day, each covering one entire year. Here, the two years are covered, including the period of the study.

We are now ready to put all the pieces together in a complete monitoring rule to detect a sleeping activity.

```
sleep:
  ex(night,
     sleep_segment |
     during(~sleep_segment <= 15min, night))
```

This rule includes two new operators (`ex` and `during`) and introduces an alternative (*i.e.,* a disjunction) to a sleep segment. First, Operator `ex`, which means 'exists', ensures that the sleep activity intersects the night time slot (there exists at least a moment in the sleep activity, which happens during the night). This results in excluding naps during the day, while allowing some flexibility (*e.g.,* the sleep may go beyond the end of the night slot). Second, the alternative to a sleep segment is some other short activity happening during the night (*e.g.,* a toilet visit). Operator `during` is used to handle such sleep interruptions (*i.e.,* negation of Rule `sleep_segment`), which do not contradict a sleep activity. Specifically, an interruption does not contradict a sleep activity if it does not last for too long (the maximum pause duration is set here to 15 minutes), and occurs during (Operator `during`) the night time slots (Variable `night`).

Finally, further analysis of the sensor datatsets revealed that some participants are occasionally not sleeping in their bedroom (typically, in their living room). To account for this situation, a new monitoring rule is introduced (definition omitted) that detects this kind of sleep segments. In doing so, this routine deviation can be monitored; the user and/or their caregiver can be informed about potential resulting health issues. This situation further illustrates the need for our iterative approach to developing monitoring rules.

5.2.2 Failure detection for bedroom sensor. The same logic as that of toilet failure and door failure is used to detect failures of the

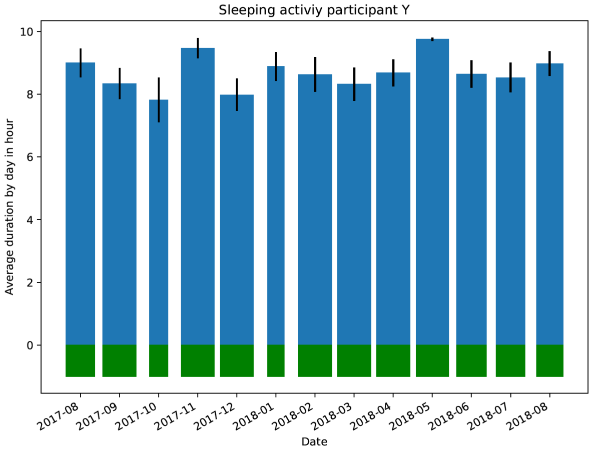

Figure 4: Visualization of sleeping activity

bedroom motion detector. Therefore, our user-defined operator may be reused.

```
bed_failure:
  sensor_failure[24hr]("MotionD_B")
```

5.2.3 Visualization. Figure 4 shows an example of visualization of data representing sleeping activity for a participant, every month during one-year period. The blue bars represent the average sleep duration of a person, which varies from 7 to 10 hours per day. We notice that the bedroom sensor has worked well, as shown by the width of the (blue and) green bars, apart from October 2017 and February 2018 where the bars are thin.

5.3 Platform failures

Our previous failure rules can detect sensor failures related to specific activities, but they do not annotate more radical failures, such as an Internet outage or a general platform failure when no sensor is working. In fact, it is important for technicians to obtain such information for maintenance purposes of the smart-home infrastructure. To address this issue, we defined a rule that detects when there is no activity in the home for a given duration, provided it is not an outing.

```
def platform_failure[T] =
  no_activity > T & ~outing_period[T]

platform_failure_1day:
  platform_failure[24hr]
```

Note that this rule reuses the one detecting outings defined in Section 5.1.

6 EVALUATION OF ACTIVITY MONITORING RULES

Using Signal Detection Theory [20] (SDT), an expert in activity analysis, served as a human judge to examine some samples from the datasets of 5 participants to manually label activities and sensor failures. Then, the manual labels were compared to the ones generated by our rules to determine whether these rules are as accurate as our human expert to detect activities. In doing so, the output of the monitoring rules were tested against the answers of the human judge, which were used as the ground truth.

Table 3: Evaluation of monitoring rules using SDT

Results Rules	Trials	Hits	False alarm	*A’*	*B”D*
Outgoing	13	13	0	1.00	0.00
Toilet	45	41	10	0.91	0.22
Sleep quiet	16	14	2	0.87	0.12
Door failure	7	7	0	1.00	0.00
Bed failure	9	9	0	1.00	0.00
Toilet failure	3	3	0	1.00	0.00
Platform failure	2	2	0	1.00	0.00

Thirty five samples of sensor logs were randomly selected from the datasets collected at participants’ home. For each participant, the samples of sensor logs covered three entire days for each activity, and an entire month for each kind of sensor failure.

Two specific indices were calculated for each monitoring rule: the sensitivity and the response bias indices, noted respectively *A’* and *B"D*. Sensitivity indices are used in signal detection theory to measure performance in Yes/No tasks. Specifically, participants are asked to discriminate against signals (the stimulus is present) and noise (stimulus is absent). In the presence of a stimulus, affirmative responses are correct and are called hits. In the absence of a stimulus, the answers ‘yes’ are incorrect and called false alarms. Then, successes and false alarms are used to calculate the indices. The sensitivity *A’* measures the participant’s ability to correctly discriminate the presence or absence of stimuli. This index is between 0 (extremely low sensitivity) and 1 (extremely high sensitivity). The response bias *B"D* measures the participant’s general tendency to answer yes or no. *B"D* is between -1 (tendency to answer yes and produce false alarms) and 1 (tendency to answer no and miss the stimuli), the ideal value of this index being 0. In our experiment, the rules take the role usually played by human participants in Yes/No tasks. These tasks are independent because they concern activities that occur at a specific location (*i.e.,* targeting specific sensors) and time periods. Thus, this labeling process amounts to a binary classification.

The complete results are shown in Table 3. For the rules concerning 1) outings, 2) platform failures, and 3) sensor failures (related to outings, visiting the toilet and sleeping activity), the values of both sensitivity and response bias are ideal: *A’* = 1.00 and *B"D* = 0.00. These results demonstrate that our rules produce correct results with respect to the human observer. The rules can be considered as extremely sensitive and fits perfectly the observer in this case.

Results pertaining to the detection of toilet visits show a very good sensitivity, *A’* = 0.91. That means that most of the responses of the rule were correct. The corresponding rule is said to be highly sensitive. Furthermore, it shows a reasonable response bias of *B"D* = 0.22, which indicates that the rule is slightly conservative, in that it misses a few stimuli (the rule has a slight tendency to respond No).

Results for sleeping activity also has both good sensitivity and response bias, *A’* = 0.87 and *B"D* = 0.12, which show that most of the rule-generated values match the judgement of the human observer. The response bias index indicates that the rule rarely misses stimuli.

To sum up, our rules are accurate in that they always detect a sensor-failure and almost always detect if an activity of interest is present in a given dataset, as compared to our human observer. As such, this evaluation contributes to validate our tool-based methodology to developing activity-monitoring rules.

7 CONCLUSION AND FUTURE WORK

This paper has presented a tool-based methodology to develop knowledge-based rules dedicated to processing longitudinal, real-world, home-centric, sensor data. The proposed approach reliably detects older adults' activities and provides professional caregivers with actionable insights via a visualization tool. To contribute to the reproducibility of our results, we have made publicly available a realistic set of sensor data and our rules to detect activities.

This work improves replicability of activity monitoring research in that it introduces an iterative process to develop concise and high-level activity-monitoring rules. The proposed approach contributes to expose and systematize the stepwise refinement of monitoring rules by making explicit this iterative process, leveraging user-specific knowledge, and abstracting over hard-to-anticipate, yet typical situations. Our methodology was illustrated by using it in a case study, which involved monitoring data of five different older adults in their respective dwellings during several months. This case study has shown the generality of our methodology, which was successfully applied across the characteristics of individuals, their routines, their home layouts, *etc.*

Using Signal Detection Theory, our rules have been shown to be accurate and reliable for detecting sensor failures and various activities (sleeping, toilet visits and outings). However, this accuracy has only been validated by a unique human expert. In the future, we plan to improve this aspect by involving three human observers to assess the interjudge reliability of the proposed approach. Furthermore, more daily activities will be considered to investigate the range of applicability of our methodology.

Another line of work to explore consists of linking a user's activity data to their clinical data (*e.g.,* sensory and motor functioning, hospitalization, frailty and degradation *etc.*) over a long period of time so that care professionals, such as occupational therapists, can evaluate potential signs of age-related decline. This work goes beyond our case study, which was limited in duration and did not consider the clinical data of their participants. Lifting these limitations could pave the way to screening capabilities, which is a driving force for the development of activity monitoring systems dedicated to older adults.

REFERENCES

[1] Valérie Bergua, Jean Bouisson, Jean-François Dartigues, Joel Swendsen, Colette Fabrigoule, Karine Pérès, and Pascale Barberger-Gateau. 2013. Restriction in instrumental activities of daily living in older persons: Association with preferences for routines and psychological vulnerability. The International Journal of Aging and Human Development 77, 4 (2013), 309–329.

[2] Loïc Caroux, Charles Consel, Lucile Dupuy, and Hélène Sauzéon. 2018. Towards context-aware assistive applications for aging in place via real-life-proof activity detection. Journal of ambient intelligence and smart environments 10, 6 (2018), 445–459.

[3] Charles Consel, Lucile Dupuy, and Hélène Sauzéon. 2017. HomeAssist: An assisted living platform for aging in place based on an interdisciplinary approach. In International Conference on Applied Human Factors and Ergonomics. Springer, 129–140.

[4] Diane Cook, M Schmitter-Edgecombe, Aaron Crandall, Chad Sanders, and Brian Thomas. 2009. Collecting and disseminating smart home sensor data in the CASAS project. In Proceedings of the CHI workshop on developing shared home behavior datasets to advance HCI and ubiquitous computing research. 1–7.

[5] Diane J Cook, Narayanan C Krishnan, and Parisa Rashidi. 2013. Activity discovery and activity recognition: A new partnership. IEEE transactions on cybernetics 43, 3 (2013), 820–828.

[6] Joëlle Coutaz and James L Crowley. 2016. A first-person experience with end-user development for smart homes. IEEE Pervasive Computing 15, 2 (2016), 26–39.

[7] Sara J Czaja, Walter R Boot, Neil Charness, and Wendy A Rogers. 2019. Designing for older adults: Principles and creative human factors approaches. CRC press.

[8] Prafulla N Dawadi, Diane J Cook, and Maureen Schmitter-Edgecombe. 2013. Automated cognitive health assessment using smart home monitoring of complex tasks. IEEE transactions on systems, man, and cybernetics: systems 43, 6 (2013), 1302–1313.

[9] Prafulla N Dawadi, Diane J Cook, Maureen Schmitter-Edgecombe, and Carolyn Parsey. 2013. Automated assessment of cognitive health using smart home technologies. Technology and health care 21, 4 (2013), 323–343.

[10] Lucile Dupuy, Charlotte Froger, Charles Consel, and Hélène Sauzéon. 2017. Everyday functioning benefits from an assisted living platform amongst frail older adults and their caregivers. Frontiers in aging neuroscience 9 (2017), 302.

[11] Nancy ElHady and Julien Provost. 2018. A systematic survey on sensor failure detection and fault-tolerance in ambient assisted living. Sensors 18, 7 (2018), 1991.

[12] Xin Hong and Chris D Nugent. 2013. Segmenting sensor data for activity monitoring in smart environments. Personal and ubiquitous computing 17, 3 (2013), 545–559.

[13] Mohammad Anwar Hossain. 2014. Perspectives of human factors in designing elderly monitoring system. Computers in Human Behavior 33 (2014), 63–68.

[14] Justin Huang and Maya Cakmak. 2015. Supporting mental model accuracy in trigger-action programming. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing. ACM, 215–225.

[15] Beth Logan, Jennifer Healey, Matthai Philipose, Emmanuel Munguia Tapia, and Stephen Intille. 2007. A long-term evaluation of sensing modalities for activity recognition. In International conference on Ubiquitous computing. Springer, 483–500.

[16] Barry Reisberg, Sanford Finkel, John Overall, Norbert Schmidt-Gollas, Siegfried Kanowski, Hartmut Lehfeld, Franz Hulla, Steven G Sclan, Hans-Ulrich Wilms, Kurt Heininger, et al. 2001. The Alzheimer's disease activities of daily living international scale (ADL-IS). International Psychogeriatrics 13, 2 (2001), 163–181.

[17] Daniele Riboni, Linda Pareschi, Laura Radaelli, and Claudio Bettini. 2011. Is ontology-based activity recognition really effective?. In 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops). IEEE, 427–431.

[18] AM Sarkar, Young-Koo Lee, and Sungyoung Lee. 2010. ARHMAM: an activity recognition system based on hidden Markov minded activity model. In Proceedings of the 4th International Conference on Uniquitous Information Management and Communication. ACM, 71.

[19] Adriana M Seelye, Maureen Schmitter-Edgecombe, Diane J Cook, and Aaron Crandall. 2013. Naturalistic assessment of everyday activities and prompting technologies in mild cognitive impairment. Journal of the International Neuropsychological Society 19, 4 (2013), 442–452.

[20] Harold Stanislaw and Natasha Todorov. 1999. Calculation of signal detection theory measures. Behavior research methods, instruments, & computers 31, 1 (1999), 137–149.

[21] An C Tran, Stephen Marsland, Jens Dietrich, Hans W Guesgen, and Paul Lyons. 2010. Use cases for abnormal behaviour detection in smart homes. In International Conference on Smart Homes and Health Telematics. Springer, 144–151.

[22] TLM Van Kasteren, Gwenn Englebienne, and Ben JA Kröse. 2010. Activity recognition using semi-markov models on real world smart home datasets. Journal of ambient intelligence and smart environments 2, 3 (2010), 311–325.

[23] Nic Volanschi, Adrien Carteron, and Charles Consel. 2018. A Domain-Specific Approach To Unifying The Many Dimensions of Context-Aware Home Service Development. In 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). IEEE, 480–489.

[24] Nic Volanschi, Bernard Serpette, Adrien Carteron, and Charles Consel. 2018. A Language for Online State Processing of Binary Sensors, Applied to Ambient Assisted Living. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2, 4 (2018), 192.

[25] Kristina Yordanova, Stefan Lüdtke, Samuel Whitehouse, Frank Krüger, Adeline Paiement, Majid Mirmehdi, Ian Craddock, and Thomas Kirste. 2019. Analysing Cooking Behaviour in Home Settings: Towards Health Monitoring. Sensors 19, 3 (2019). https://doi.org/10.3390/s19030646