A Tooled Method for Developing Knowledge-Based Activity Recognizers

Rafik Belloum, Antoine Riche, Nic Volanschi, Charles Consel

A Tooled Method for Developing Knowledge-Based Activity Recognizers

Rafik Belloum
Télécom Paris & Inria, France
rafik.belloum@telecom-paris.fr

Antoine Riche
Inria, Bordeaux, France
antoine.riche@inria.fr

Nic Volanschi
Inria, Bordeaux, France
nic.volanschi@free.fr

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

***Abstract*—Monitoring the daily activities of older adults is a key enabler for aging in place because it reliably indicates whether autonomy is preserved and it prevents unwanted situations (*e.g.,* lack of activity during daytime). To fulfill its promises, activity monitoring requires development methods capable of systematically delivering activity recognizers that are accurate enough to be trusted and accepted by users and their caregivers.**

This paper presents a systematic approach to developing accurate activity recognizers, based on a tooled method. To achieve accuracy, our strategy is twofold: 1) to encompass the main variations of a target activity by abstracting over descriptions reported by users; 2) to ensure proper customization with respect to user specificities using a dedicated tool. This development method is iterative, allowing to adjust the parameters of an activity recognizer to maximize its accuracy.

We validated our approach by applying it to a case study. Specifically, we applied our tooled method to the development of 6 generic activity recognizers, which were then customized with respect to the specificities of 5 older adults, and deployed in their homes during 5 days. Once deployed, the results produced by these activity recognizers were checked daily against activities self-reported by our participants. This experiment shows that 80% of the outputs of our activity detectors were confirmed by the user reports. The accuracy of our approach goes up to 88% when considering the four, more routinized participants.

***Index Terms*—human activity recognition, ambient assisted living, smart homes, agile methodologies, aging in place**

I. Introduction

The aging of the population raises a vast societal challenge to support the needs of older adults and to enable them to live independently. To address this challenge, a promising approach revolves around assistive computing and consists of equipping the home of older adults with pervasive computing technologies and services dedicated to monitoring and assisting their daily activities [1]. In this approach, it is paramount to monitor the daily activities performed by older adults, such as a bedtime routine and meal preparation, because they give a reliable indication of whether autonomy is preserved and prevent unwanted situations (*e.g.,* lack of activity during daytime). In particular, the goal of this monitoring is to assess how daily activities evolve over time. On the one hand, a sudden surge in activity misses is valuable information for a caregiver or a health professional that can result in a prompt intervention. On the other hand, a steady increase in activity misses is useful for a caregiver to anticipate compensation measures (*e.g.,* a meal delivery service).

To fulfill its promises, activity monitoring requires development methods capable of systematically delivering activity recognizers that are accurate enough to be trusted and accepted by users. Indeed, considering how much they are to be intertwined in the daily life of users, activity detectors with low accuracy may do more harm than good. For example, consider an activity recognizer that falsely misses activities and issues erroneous reminders to a frail user and their caregiver. At best, such a service is quickly ignored and/or unplugged by the user; at worse, it may have a deleterious effect on them.

The major challenge when developing activity recognizers is to make them both generic and specific: generic to cope with a wide range of home configurations and user routines, and specific to detect activities with a sufficient level of accuracy. Black-box approaches based on machine learning are very powerful for dealing with a potentially high volume of data and delivering statistically correct answers. However, such answers may not be predictable enough, nor easy to explain to older adults and caregivers. Moreover, approaches based on machine learning require a great amount of training data (sometimes tagged by experts) in order to be effectively specialized for each home/user configuration. These limitations suggest that deterministic solutions to activity recognition are better suited for assistive computing and that activity recognizers should be both generic to scale, and customizable to account for user/home specificities.

Recently Caroux *et al.* proposed an approach to daily activity recognition for older adults that aims to verify daily routines based on user declarations [2]. Specifically, daily routines are initially declared by users and their caregivers; these declarations are then formalized into simple formulas, which model user interactions with their environment. Formulas, generalized across users, are matched against sensor data to determine whether daily routines have been performed. Because their approach is driven by user declarations, activities are verified and not inferred, delivering predictable information. Although, activity verification has shown promising results [3], developing activity recognizers involves ad hoc and manual steps that prompt a need for methodological and tool support.

This paper presents an agile method dedicated to producing accurate and rapidly customizable activity recognizers. Our method is 1) knowledge based in that it involves declarations from users and their caregivers to drive the service customization process, leveraging Caroux *et al.*'s approach, and 2) data centric in that it uses real sensor logs from smart homes, untagged and in small amount, to achieve the required level of accuracy.

Our work makes the following key contributions.

- We present a disciplined method to develop *generic* activity recognizers. This approach leverages user declarations and generalizes over inter-individual variabilities.
- We develop a visualization tool used by our method to allow *rapid customization* of generic activity recognizers.
- We expand the range of target activities, compared to Caroux *et al*, and generalize their formula-based approach to cope with partially performed activities.

We validate our tool-based method by measuring the accuracy of our set of activity recognizers in a case study on 5 users and 6 activities, namely, bedtime routine, wakeup routine, outings, preparation of breakfast, lunch and dinner.

The rest of the paper is strucured as follows. Section II presents the related work and Section III provides some background information and context for the proposed work. Section IV defines our disciplined and tool-supported method. Section V applies our method to the development of 6 customizable activity recognizers, tested by 5 older adults in their home. Section VI assesses the accuracy of the developed activity recognizers by matching their results against the user truth. Section VII discusses our results and Section VIII presents concluding remarks.

II. Related work

Homes are rapidly becoming connected with the deployment of pervasive computing technologies, consisting of conventional devices, such as motion detectors, contact sensors and connected plugs, and emerging technologies, such as smart speakers, learning thermostats, connected door locks. Today, most of these technologies automate a narrow range of tasks, such as voice-activation of appliances and heating system driven by machine learning. Tomorrow, pervasive computing, scaled up by the Internet of Things, will allow to maintain health and independence of older adults [4]. Taking up this challenge, researchers have been developing smart homes and services to monitor older adults, and study how age decline impacts their cognition and everyday functioning. A major project focusing on the monitoring of older adults is CART [5], where longitudinal, naturalistic, observational cohort studies are conducted at a large scale (totaling over 400 participants). Other smart home-based projects for aging complement the monitoring with services that assist older adults in their daily activities. This approach is pursued by HomeAssist [6], which provides assistance to older adults in the form of notifications to remind them of an activity (*e.g.,* an appointment) and to alert them about an undesirable situation (*e.g.,* a door left open). Regardless of the approach, smart homes for aging in place revolve around activity recognition, that is, the ability to recognize human activities [7]. Because of privacy and intrusiveness concerns, activity recognition for older adults often precludes the use of cameras when studies are conducted in their homes [8]. As a consequence, in the context of a smart home, most approaches to activity recognition rely on sensors that detect interactions of the older adult with their environment (*e.g.,* a door opened, a motion in a room) [5], [6].

In a real-life setting, recognizing human activities needs to account for a range of inter-individual variabilities. These variabilities include the home features, a caregiving context, user specificities, requirements, and preferences. Neglecting these variabilities can result in using an inappropriate sensor, misplacing it, or making a false assumption. In turn, this leads to missing user activities or misinterpreting sensor data. When studies are conducted over a long period of time, collecting incomplete or incorrect sensor data can turn into a vast waste of time and energy. The range of variabilities in real-life settings and their unexpected nature have been a major barrier for the applicability of activity recognition approaches based on machine learning and activity models. This difficulty is illustrated by Dawadi *et al.* that restrict their use of machine learning to infer activities in a lab setting [9].

Caroux *et al.* introduce an alternative to inferring activities, named *activity verification* [2], [3]; it is inspired by Chen *et al.*'s knowledge-based approach [10] and has been successfully applied to a real-life setting: homes of older adults. Activity verification leverages knowledge about users to verify their daily activities; the verification is driven by the characteristics of the user and their daily activities. Activity verification targets older adults because these individuals are known to routinize their daily activities as they age [11]. Caroux *el al.* use declarations provided by older adults to model their activities. In doing so, a user declares the characteristics of each activity of interest. Specifically a user is asked to situate the activity in a room (*i.e.,* where), to identify the user-environment interactions (*i.e.,* how), and to give a time at which the activity occurs (*i.e.,* when). The observed user-environment interactions give a list of *markers* that characterize an activity (*e.g.,* breakfast preparation involves turning on the coffee machine, getting a mug from a kitchen cabinet, taking a milk bottle from the fridge). Markers are not equally reliable to detect an activity: some are said to be *primary markers* because they are present every time the activity is performed (*e.g.,* coffee machine); whereas others are said to be *secondary markers* because they may sometimes be missing (*e.g.,* a mug can be taken from the dishwasher, instead of the usual cabinet). In practice, activity verification requires a minimal set of sensors because markers have been carefully selected based on the user-declared routines. Furthermore, the approach only requires three kinds of sensors: motion detectors (room presence), contact sensors (room/entrance and cabinet doors) and connected plugs (appliance usage). Their placement is driven by user declarations to target specific user-environment interactions. Although promising, Caroux *et al.*'s approach involves ad hoc and manual steps to achieve accuracy.

Our work takes activity verification further by systematizing and tooling the development of accurate activity recognizers. Achieving accuracy is driven by a multi-step development method that leverages user declarations to generalize over inter-individual variabilities while ensuring proper customization with respect to user specificities. This development method is iterative and allows to adjust the parameters of an activity recognizer to maximize its accuracy.

III. Background

To develop our proposed method, we leveraged the HomeAssist project [6], which aims to support aging in place by developing and deploying a smart home platform in the home of older adults. This platform consists of sensors, which provide contextual information to a set of assistive services, and actuators, which allow these services to take actions, if needed. These services target three assistive domains: 1) they monitor activities of daily living and providing assistance when necessary (*e.g.,* reminders, task prompting); 2) they alert the user and/or caregiver when security issues are detected (*e.g.,* entrance door left open); 3) they support social interactions (*e.g.,* collaborative games). The HomeAssist platform was used in a field study and deployed in over 140 homes of older adults, aged 80 years and older, living alone, during a maximum of 24 months. This field study revealed the positive impact of HomeAssist on participants in terms of daily autonomy, self-regulation and empowerment [12].

For each participant, depending on their needs, specific activities are targeted for assistance. Declaring an activity includes having the user sketch the activity of interest in their home to determine reliable markers. Table I presents a typical list of sensors deployed in a home; the first column lists the rooms fitted with sensors, whose names are defined in the second column (Sensor ID) – these names are later used to discuss activity recognizers. The last column of Table I defines the function for each sensor deployed in a home, that is, the meaning of the sensor measurements.

IV. Development Method

This section defines our disciplined and tool-supported method for the agile development of activity recognizers. The overall view of our approach is depicted in Figure 1. Let us examine the key concepts and steps, forming our approach.

Step 1 of our approach (noted "1. Declaration" in Figure 1) is the declaration of routines by the seniors and/or their caregivers (noted '(a)' in Figure 1). During interviews using dedicated questionnaires, they declare the steps used to perform their daily routines (*e.g.,* "When I wake up, I come out of the bedroom and shortly afterwards I go to the kitchen") and they provide estimated values for these steps (*e.g.,* bedtime,

TABLE I
HomeAssist sensors and their functions

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

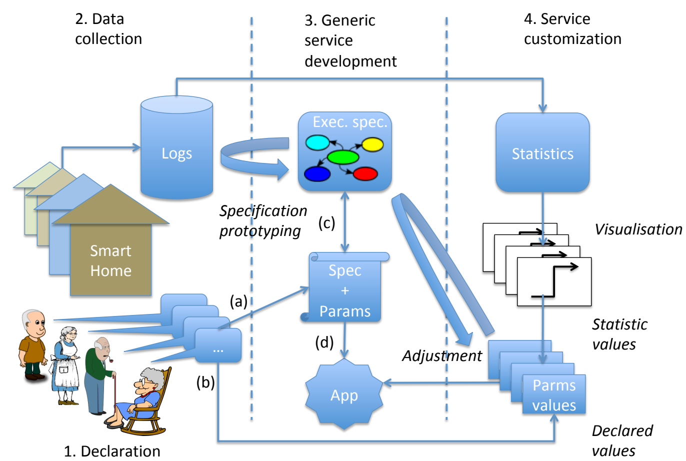

Fig. 1. Overview of the development method.

wakeup time, transition time between bedroom and kitchen in the morning). These values serve as parameters for service customization (noted '(b)' in Figure 1).

Step 2 of our approach is the deployment of sensors in the home of a senior, and the collection of the logs during a setup period. Only the sensors required to verify the routines declared at Step 1 are installed. In our method, the logs gathered during the setup period are used as a base line to build and tune the target activity recognizers.

Next, Step 3 consists of iteratively developing a service to recognize a target activity. The service needs to be generic enough, not only to cover all the declared variations of user routines, but also to cope with diverse home configurations. Indeed, homes may range from small apartments to houses with several floors. The agility of this development step hinges on a rapid prototyping cycle. However, from our experience developing a range of assistive services, a prototyping cycle for simple activity recognizers takes in the order of 2 person-weeks, assuming a general-purpose programming language is used, such as Java, as well as state-of-the-art development tools. Additional time is also needed to deploy and test activity recognizers in the homes of older adults. The duration of this process does not meet our requirement of rapid iterative prototyping of generic services.

To resolve this issue, we chose to raise the level of abstraction at which activity recognizers are developed by using a scripting language. Furthermore, we decided to prototype activity recognizers by running them against recorded logs, instead of deploying them. In practice, our strategy is particularly well-suited for developing and testing the kind of applicative logic needed to detect daily activities declared by users. Indeed, daily activities consist of events (*e.g.,* motion detected, door closed) with ordering constraints and time delays that can naturally be expressed as timed automata, which are to be matched against event logs. As such, developing activity recognizers requires a programming language with limited but specialized expressive power. To ease the prototyping process, event logs are kept in a simple textual format (JSON format, with one sensor event per line), ensuring good readability for easy manual inspection, understanding, and debugging. Considering the textual nature of the data to be processed, we chose Perl as the scripting language to benefit from its rich set of text processing operations.

Assessment of our new strategy revealed that the Perl-scripted, executable specifications of activity recognizers incurred a development cycle of less than 1 person-day. As such, it is short enough to be considered an agile iterative development.

As a specification gets tested against an increasing number of logs, coming from different homes (noted '(c)' in Figure 1), its generality typically grows by introducing new parameters; *e.g.,* delays between user actions or the name of the room where the user sleeps at night. Once the Perl-scripted specification covers all the configurations, it is implemented as a generic service over the sensor infrastructure, using appropriate technology (noted '(d)' in Figure 1). In our case, we use the Java programming language and the HomeAssist platform [6], [13].

Finally, Step 4 is the generic service customization for each configuration. In principle, parameter values can simply be extracted from user declarations. However, as we show in Section VI, values provided by users are only estimates, and can rarely be used as final customization values. Thus, it is necessary to check these values against real logs, and perform the necessary adjustements. Considering the potentially large number of user/home configurations for a given service (over 100 homes in the HomeAssist project), it is critical to use an efficient process to find the right parameter adjustments. This is why we developed a visualization tool for assisting this instantiation step. More specifically, this tool performs statistic analyses on the logs and displays the results in a visual form as histograms to facilitate the manual validation or adjustment of parameter values for the generic services. The histograms allow one to understand the typical values for a given home/user configuration. For example, the time slots and the appropriate appliances for detecting a lunch activity in a specific user-home configuration can be found using histograms of appliance usage from log data. Also, the correct threshold for the delay between the wakeup time of a user and the start of their morning routine can be easily observed using an appropriate histogram.

Whenever a set of values is chosen for the parameters of an activity detector during Step 4, the visualisation tool allows to execute the scripted specification of the detector on any smart home log, and to display the results as a list of detected activities for each day. This allows to instantly see the effect of changing a parameter value and relate this value with the one declared by the user.

Thus, our visualization tool enables rapid customization decisions based on automated statistic analyses and dedicated display functionalities.

V. Case study

We now present the case study[1] used to validate our approach. Specifically, we applied our tooled method to the development of 6 generic activity recognizers, which were then customized with respect to 5 older adults, and deployed in their homes during 5 days. Once deployed, the results produced by these activity recognizers were checked daily against activities self-reported by our participants. Let us describe each step of our study.

A. Declaration and data collection

The declarations of activities of interest were gathered at the installation time of the platform in each home. Activities were declared using questionnaires dedicated to extract key inputs for the sensor-based verification process. For instance, for the preparation of each meal during the day, the older adult and/or caregiver were asked to provide the approximate time period of this activity and the appliances used to perform the task. After gathering user declarations, sensor logs started to be accumulated and provided a basis to assess their accuracy.

B. Generic service development

Developing generic services is driven by initial declarations of older adults and their caregivers describing the steps involved in performing the activities of interest. Analyzing the inter-individual variations is essential to determine where genericity (*i.e.,* parameters) is needed to abstract over these variations. As illustrated with the activity recognizers presented below, the parameters often need to be adjusted when applied to real homes and users. Note that we only discuss parameters that are not self-explanatory.

Each activity recognizer is briefly described. Its behavior is then formalized in the form of an automaton. Finally, the

[1]The present case study is publicly available at the following URL: https://github.com/belloum/data-analyser

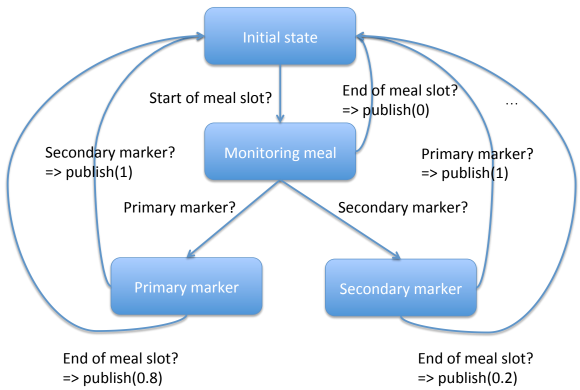

Fig. 2. Automaton for recognizing the different meal routines.

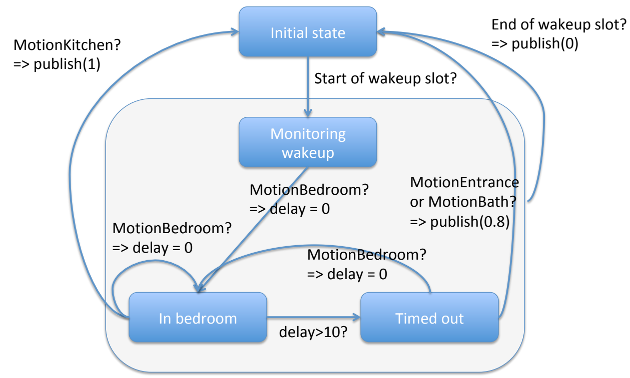

Fig. 3. Timed automaton for recognizing the wakeup routine.

list of its parameters are presented, as well as its evolution to capture unanticipated variations.

1) Meal preparation: In pursuit of genericity, we set out to develop one service that could cover all three meals (breakfast, lunch and dinner), as opposed to one service for each meal.

a) Logic: The logic of this service is implemented by the automaton in Figure 2. The automaton starts in the initial state and begins monitoring a meal when the corresponding time slot starts (transition to the "monitoring meal" state). While in this state, the service waits for the markers associated to the meal to be detected. If the primary marker is detected first, another state is reached where the secondary marker is waited for, and vice versa. If both are detected, in any order, the automaton publishes a value of 1, meaning that the meal has been definitely recognized, and resets itself. If, however, the end of the meal slot happened before the sequence is complete, the automaton resets itself without waiting further markers. Depending on the markers already seen (which are encoded in the current state), the published value may be 0 (meaning that no meal has been recognized), 0.2 (meaning that only the secondary marker has been activated), or 0.8 (meaning that only the primary marker has been activated).

b) Initial parameters: meal name, time slot, primary marker, secondary marker.

c) Added parameters: several primary markers, several secondary markers. In a second iteration of our method, the parameters for the markers had to be extended from a single sensor to a list of sensors. This is because, for some participants, there are variants of meal preparation that need to be covered by a set of primary (or secondary) markers, from which any appliance is considered part of the activity. For instance, a participant may prepare breakfast using either the coffeemaker, the microwave, or the fridge as a primary marker, while the cupboard door detected open is always the secondary marker.

2) Wakeup routine: The wakeup routine detects a user starting their day. The challenge is to exclude situations where the user wakes up during the night to visit the toilet or to drink in the kitchen and later goes back to bed. The two key elements to consider are the time period at which the user normally wakes up and how much time it takes them to go to the kitchen to start their day.

a) Logic: The logic of this service is implemented by the *timed* automaton in Figure 3. Indeed, with respect to the previous service recognizing meals, the present service has to check some timing constraints. A timed automaton is adequate for this purpose, as it contains *clock variables* that may be reset by transition actions and may be read by the transition conditions. The automaton transitions from the initial state to a monitoring state when the wakeup time slot starts. Subsequently, when user motion is detected in the bedroom, the 'delay' clock variable is reset and a transition is taken towards state "In bedroom". Any further motion in the bedroom resets this clock. Upon a motion in the kitchen, a value of 1 is published, which corresponds to a full recognition of the wakeup routine. Alternatively, if the clock reaches 10 minutes, the "timed out" state is reached. While in this timeout state, a value of 0.8 may be published (meaning that the routine has been partially recognized) if motion is detected in a different room, excluding the kitchen. Alternatively, a new motion in the bedroom triggers a transition back to the previous state "In bedroom". However, if the end of the wakeup time slot is reached, no matter the current state, the automaton resets itself and publishes a value of 0 (meaning that the wakeup routine was not detected at all). This is expressed by the transition originating in the compound state regrouping all the previous three states.

b) Initial parameters: time slot, delay from bedroom to kitchen.

c) Added parameters: room where the user usually sleeps (default: bedroom), room where activity occurs in the morning (default: kitchen). Indeed, a second iteration of our

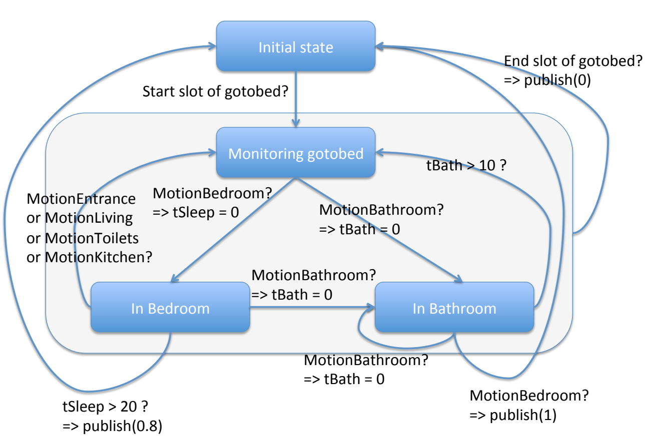

Fig. 4. Timed automaton for recognizing the bedtime routine.

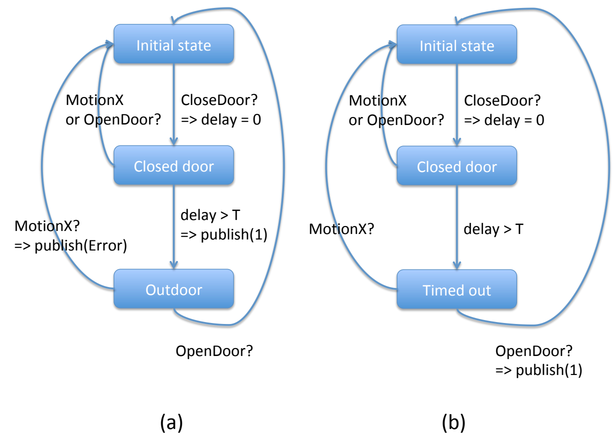

Fig. 5. Timed automaton for recognizing outings: (a) in real time; (b) *a posteriori*.

method consisted in allowing to parameterize the rooms for the wakeup routine, as they are not always the bedroom and the kitchen. For instance, after wakeup, a participant may start their day by visiting the bathroom to shower, rather than the kitchen to prepare breakfast.

3) Bedtime routine: This routine targets the actions performed by a user before going to bed at a specific time period. The typical pattern we considered is a visit to the bathroom shortly followed by an extended stay in the bedroom.

a) Logic: The logic of this service is implemented by the timed automaton in Figure 4. Starting in the initial state, a transition to a monitoring state is taken upon the start of the indicated time slot. Here, a motion in the bathroom or in the bedroom causes a transition to one of two states: "In bathroom"' and "In bedroom", respectively. A corresponding clock variable, tSleep or tBath, is also reset. While in the bathroom, any motion in the bedroom within 10 minutes (for instance) causes a full recognition of the routine (by publishing a value of 1). While in the bedroom, any motion in the bathroom transitions to the previous state "In bathroom"; any motion elsewhere causes a transition back to the monitoring state. Alternatively, if no movement is sensed anywhere else for 20 minutes (for instance), the routine is partially recognized (by publishing a value of 0.8). However, if the end of the given time slot is reached, whatever the current state, the automaton resets itself without recognizing the routine at all (by publishing a value of 0).

b) Initial parameters: time slot, delay from bathroom to bedroom.

c) Added parameters: room where activity occurs last (default: bathroom), room where the user usually sleeps (default: bedroom). Indeed, in a second iteration of our method, the rooms involved in the bedtime routine were parameterized because they are not always the bathroom and the bedroom. For instance, a participant may visit the toilets, rather than the bathroom, shortly before going to bed.

4) Regular outings: This service detects when the user departs from home to conduct some activity outside. The key insight to detect an outing is to monitor the entrance door and motion within the home.

a) Initial logic: The initial logic of the service was implemented by the timed automaton in Figure 5 (a). This automaton recognizes an outing soon after a closed door, if the door is not opened and no motion in the home has been sensed during a certain delay T. For that, it uses the clock variable 'delay'. When the clock reaches T, the automaton signals the outing and goes to the 'Outdoor' state. In this state, the only legal transition is when the door is opened, which signals the end of the outing. Sensing any motion within the house in this state, before opening the door, means that the person was really inside the house, so the signalled outing was in fact a false positive. Therefore, an error is published to signal the mistake. We initially thought that choosing a suitable value for the delay T should cover all possible configurations. But in fact, this specification never worked reliably in all the homes with any reasonable delay: the service sometimes raised errors. This is because the motion detectors of many homes do not exhaustively cover the space. Thus, it is possible for the user to close the door from the inside, and stay undetected inside the home for an arbitrary long time.

b) Final logic: In a later iteration of our method, after having tried different designs, we aimed to detect outings in an *a posteriori* way. Specifically, an outing is signalled when no motion has been sensed within the home since the entrance door was closed *and until it is opened again*. This logic is implemented by the automaton in Figure 5 (b). Note that, in this version, the value of 1 (signalling the outing) is not published until the entrance door is opened again. The delay T in this case only serves to signal outings lasting more than T. This parameter can be set to any value (*e.g.,* filtering out short outings for checking the mailbox or emptying the thrash) without incurring any risk of creating false positives. Although accurate in practice, this approach comes at a price: outings are never detected in real time, but only when the user returns home.

c) Parameters: minimum duration of the outings (value of T).

C. Service customization

Once developed as automata scripted in Perl, the generic services can be customized, leveraging our visualisation tool. We illustrate the customization steps on one user/home configuration for all the above 6 services.

In a first phase, the visualization tool is used to produce histograms of the various sensors events in a log, distributed across a 24-hour period. These histograms provide a graphical summary of events gathered during the whole setup period, spread across a single day representation. Figure 6 displays three such histograms, one for each category of sensors deployed in a home: contact sensors, electric meters, and motion detectors. In each histogram, time is placed in the X-axis, containing 24 labels representing a day (of 1 hour length in the figure, but other granularities can be chosen), and, in the Y-axis, the number of events is placed, computed from the log for each sensor within a given hour. In our case study, the logs cover a setup period of two weeks. Note that using logs of several weeks provides confidence in the activity patterns revealed by our visualization tool.

From these histograms, one can observe that the peaks in the opening of the fridge and the cupboard occur around the time meals are being prepared. Consequently, these events can be used either as primary or secondary markers for preparation activities of the three meals of the day. In contrast, the toaster was only used twice during the two-week period at breakfast time; the microwave was only used once, in the afternoon. From these occurrences, one can conclude that the toaster is used rather rarely and, if used as a marker for breakfast preparation, it needs to be combined with some other marker to be reliable.

At this point, we have identified initial candidates for primary and secondary markers of our service recognizers. Furthermore, initial candidates for the time periods of activities can be extracted from user declarations. These candidate configuration parameters allow to make an assessment of the accuracy of the service recognizers, by executing the scripted service recognizers on the log. Table II shows, for each of our service recognizers, the initial and adjusted/final customization settings. Each customization is assessed and its success rate in recognizing the target activity is reported in the last column of the table.

Let us examine Table II for a specific service recognizer: breakfast preparation. The participant declared preparing this meal between 8:30 and 9:00, by opening the fridge, using the toaster, and usually also opening the cupboard. Before even running the service recognizer, observing Figure 6, for the related event sensors and the declared period, reveals that this period is too restrictive. Indeed it does not include the peak of the fridge uses in the morning, nor the two uses of the toaster. Let us inspect the initial configuration of the breakfast preparation service, parameterized with the user declared parameters (see the top entry of Table II): time slot = 8:30-9:00; primary markers = Toaster and Fridge; secondary markers = Cupboard.

For such a configuration, the service recognizer only detects breakfast preparation in 48% of the days within the two-week setup period. This situation illustrates the typical discrepancy between user declarations and measured activities: the user information is correct overall but often inaccurate. By adjusting the time period of breakfast preparation to better reflect the measured activities (*i.e.,* setting the time interval to 07:00-09:00), breakfast preparation is recognized in 90% of cases for this period.

Similar adjustments were done to parameter values of other activity recognizers. The fixed values of these are displayed in italics in Table II. As can be seen, the time slots of *all* the recognizers had to be adjusted for this user/home configuration.

Moreover, some additional parameters had to be changed for the recognizer of the wakeup activity. Indeed, using user declarations for this routine, the detection rate was only 10%. Our participant declared to wake up between 06:00-07:00 and to go to the kitchen within 10 minutes after that (second-to-last activity in Figure II). By studying the histograms of the motion detectors (bottom part of Figure 6), we observe that motion in the bedroom (first bar of each group) is rarely followed, within less than 1 hour, by motion in the kitchen (4th bar of each group). Instead, motion in the shower (3rd bar of each group) within the following hour appears to be much more correlated to the presence in the bedroom. To account for this situation, the value of the first room, where activity occurs after waking up, was changed from kitchen to shower. Re-executing the service, with this room value and an increased delay of 1 hour, confirms this fact because the detection rate of the wakeup routine raises from 10% to 80%.

D. Testing in silent mode

We implemented the automata corresponding to the final specifications of each activity recognizer using Java, combined with the DiaSuite, a middleware underneath HomeAssist and dedicated to develop pervasive computing applications [13]. We then customized these services with respect to 5 different users and their home, leveraging the parameter values found in the previous step. Then, we deployed the services in those homes for two weeks in ‘silent mode’; that is, they ran on the real sensor infrastructure, detected their target activity, but no action was performed in response to detection or absence of the target activity (*e.g.,* no notification

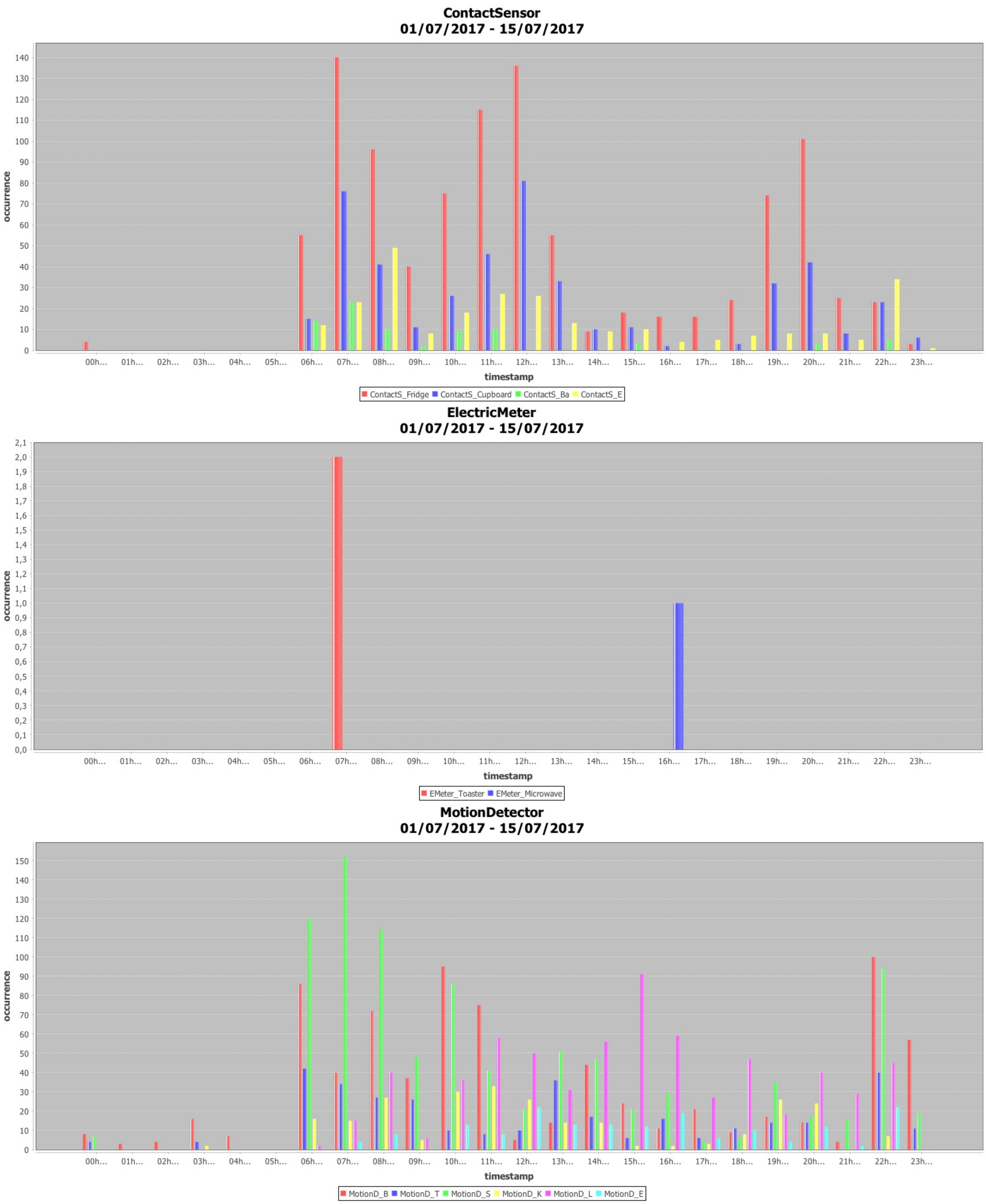

Fig. 6. Histograms of sensors spread over a 24-hour period in a home: contact sensors (top), appliance uses (middle), motion sensors (bottom).

TABLE II

EXAMPLES OF SERVICE CUSTOMIZATION FOR ONE HOME/USER CONFIGURATION.

Activity	Iteration	Slot		Parameters		Success
		Begin	End	Primary	Secondary	(%)
Breakfast	Initial config	08:30	09:00	Toaster, Fridge	Cupboard	48
	Final config	*07:00*	09:00	Toaster, Fridge	Cupboard	90
Lunch	Initial config	12:00	13:00	Fridge, Mwave	Cupboard	71
	Final config	*11:00*	13:00	Fridge, Mwave	Cupboard	81
Dinner	Initial config	19:30	20:00	Fridge, Mwave	Cupboard	57
	Final config	*18:30*	*21:00*	Fridge, Mwave	Cupboard	86
				Active room	**Delay (min)**	
Wakeup	Initial config	06:00	07:00	Kitchen	10	10
	Final config	06:00	*08:30*	*Shower*	*60*	80
Gotobed	Initial config	22:30	23:00	Shower	10	50
	Final config	*22:00*	*23:30*	Shower	10	85

issued if activity is missed). This mode only logs the detected activities.

At the end of the silent-mode period, to further ensure the reliability of the Java implementation of the service recognizers, we tested whether they behaved the same as their Perl-scripted counterparts. To do so, we checked that they detected the same activities on the collected logs.

VI. VALIDATION

Despite our test process, activity recognizers still need to be validated by their respective user to determine whether they agree on the reported activities. Filming the user around the clock in their home would be an effective approach to establishing ground truth for our services. However, the vast majority of participants rejected the option of including cameras in the set of sensors to be deployed in their home. As an alternative, the user could decide whether they agree with the detected activities; with no cameras, this approach seems to be the ultimate measure of the accuracy of our activity recognizers, and more generally, of the services produced by our tooled method.

To achieve this user-validated accuracy, we activated our services in the home of 5 users, who agreed to evaluate them during 5 days. This evaluation took the form of a questionnaire submitted daily to our 5 participants. More specifically, a questionnaire was sent every morning by e-mail to each user; it consisted of the list of activities detected the day before. The user was asked whether they approved or disproved each item of the list.

Note that, because Participant B was bedridden during our study, the services to detect wakeup and outing were not installed. Note also that a fault of the entrance door sensor, starting the very first day prevented us from installing the outing detector in the home of Participant D.[2]

An overall view of the validity of our activity detectors is shown in Figure 7. The detailed counts of valid and invalid reports for each detector are given in Table III.

[2]We chose not to visit the participant to fix the sensor during the test week in order to avoid any bias with respect to the other participants.

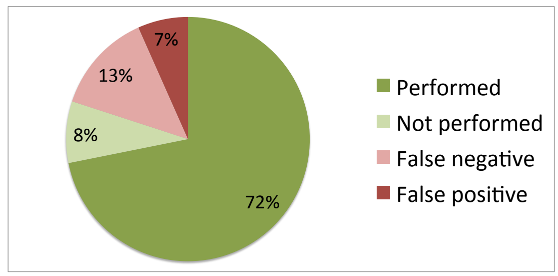

Fig. 7. Overall validity of the detected activities.

VII. DISCUSSION

As shown by Figure 7, 80% of the activity detector outputs are confirmed by the users reports: 72% of activities performed and detected and 8% of non-performed and non-detected activities. The reminding 20% correspond to wrong results produced by the activity detectors, assuming that the user responses are the ground truth. In particular, in 13% of cases, the activity was performed but slipped undetected, which can be considered a false negative, while in 7% of cases, the activity was not performed but was wrongly detected, which can be considered a false positive. Although some proportion of technical errors cannot be excluded (*i.e.,* missed events from sensors, or spurious sensor activations), we hypothesize that false negatives essentially correspond to activities that were performed by deviating from the declared routine, and that false positives were due to other activity in the home that accidentally triggered the same event patterns. This hypothesis may easily explain why false negatives are encountered significantly more often (twice as often, in our case) than false positives.

The more detailed figures in Table III show that some activity recognizers perform better than others. The least accurate recognizers are those for the wakeup and lunch activities. By diving into the details in data collected from our 5 participants for these detectors, other interesting patterns are revealed. Namely, the wrong results produced by these two detectors are specific, with very few exceptions, to Participant C (a single exception for the wakeup detector, and only two

TABLE III
DETAILED VALIDITY RESULTS FOR EACH ACTIVITY DETECTOR.

Detector	OK	KO	%
Wakeup	14	6	70
Breakfast	21	4	84
Lunch	19	6	76
Dinner	20	5	80
Gotobed	20	5	80
Exit	13	2	87

exceptions for the lunch detector). This tends to indicate that these detectors are less effective for this user. Moreover, *all* the detectors seem to be less effective on this user/home configuration, because Participant C gathers most detection errors: 14 errors. This is more than the total of 13 errors on all the other participants; these errors correspond to an overall accuracy of 53% for Participant C *vs.* an average of 88% for the other four participants. This lack of effectiveness for Participant C could be attributed to different factors, such as more routine variations, less structured time periods for activities, or technical issues with the infrastructure of this home. No matter the reason, for this case, this uneven error distribution between 80% of the users and the remaining 20% could indicate that our activity recognizers (that is, based on declarations and developed using our method) are highly adequate for most user/home configurations, and much less adequate for the remaining ones. A study on a larger sample would be needed to confirm or infirm this hypothesis.

In any case, achieving 100% accuracy in the domain of activity detection seems out of reach considering the contingencies that need to be taken into account when monitoring real users in real homes, even when users are routinized with age decline. In absence of such ideal oracles, obtaining an overall accuracy of 80% should provide older adults and their caregivers valuable information to support independent living.

VIII. CONCLUSION AND FUTURE WORK

We have presented a tooled method to develop accurate activity recognizers, which support aging in place. User declarations of daily activities are refined with sensor logs, visualized with a dedicated tool. Perl is used to rapidly script activity recognizers, which are executed over sensor logs. Then, Perl-scripted activity recognizers are implemented in Java and deployed in the homes of older adults.

We conducted a case study to put our method to practice. We scripted 6 activity recognizers, which, once refined, were implemented in Java. These services were deployed in the home of 5 older adults in silent mode (*i.e.,* without user notifications) at first to check their consistency with respect to their Perl counterparts. Once their consistency was validated, they were put in production mode. To assess the accuracy of the developed activity recognizers, their outputs were compared to the activities self-reported by our participants over a period of 5 days. This experiment shows that 80% of the outputs of our activity detectors were confirmed by the user reports. The accuracy of our approach goes up to 88% when considering the four, more routinized participants.

To further this work, our study needs to be conducted with a larger group of participants to determine whether our results scale up.

To go beyond the use of general-purpose scripting languages, we are developing a domain-specific language (DSL) with precise and formal semantics, dedicated to expressing activity detection logic. The notations and concepts of this DSL result from an analysis of existing assistive applications. This DSL should enable an even shorter development cycle in the order of 1 hour-person. A preliminary version of this DSL has already been developed and is showing promising results.

REFERENCES

[1] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 790–808, 2012.

[2] L. Caroux, C. Consel, L. Dupuy, and H. Sauzéon, "Verification of daily activities of older adults: a simple, non-intrusive, low-cost approach," in *Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility*. ACM, 2014, pp. 43–50.

[3] L. Caroux, C. Consel, L. Dupuy, and H. Sauzeon, "Towards context-aware assistive applications for aging in place via real-life-proof activity detection," *Journal of ambient intelligence and smart environments*, vol. 10, no. 6, pp. 445–459, 2018.

[4] C. Consel and J. A. Kaye, "Aging with the Internet of Things," *The Bridge*, vol. 49, no. 1, pp. 6–12, Mar. 2019.

[5] J. A. Kaye, S. A. Maxwell, N. Mattek, T. L. Hayes, H. Dodge, M. Pavel, H. B. Jimison, K. Wild, L. Boise, and T. A. Zitzelberger, "Intelligent systems for assessing aging changes: home-based, unobtrusive, and continuous assessment of aging," *Journals of Gerontology Series B: Psychological Sciences and Social Sciences*, vol. 66, no. suppl_1, pp. i180–i190, 2011.

[6] C. Consel, L. Dupuy, and H. Sauzéon, "Homeassist: An assisted living platform for aging in place based on an interdisciplinary approach," in *International Conference on Applied Human Factors and Ergonomics*. Springer, 2017, pp. 129–140.

[7] J. K. Aggarwal and M. S. Ryoo, "Human activity analysis: A review," *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, pp. 1–43, 2011.

[8] M. A. Hossain, "Perspectives of human factors in designing elderly monitoring system," *Comput. Hum. Behav.*, vol. 33, p. 63–68, Apr. 2014. [Online]. Available: https://doi.org/10.1016/j.chb.2013.12.010

[9] P. N. Dawadi, D. J. Cook, and M. Schmitter-Edgecombe, "Automated cognitive health assessment using smart home monitoring of complex tasks," *IEEE transactions on systems, man, and cybernetics: systems*, vol. 43, no. 6, pp. 1302–1313, 2013.

[10] L. Chen, C. D. Nugent, and H. Wang, "A knowledge-driven approach to activity recognition in smart homes," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 6, pp. 961–974, 2011.

[11] V. Bergua, J. Bouisson, J.-F. Dartigues, J. Swendsen, C. Fabrigoule, K. Pérès, and P. Barberger-Gateau, "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*, vol. 77, no. 4, pp. 309–329, 2013.

[12] L. Dupuy, C. Froger, C. Consel, and H. Sauzéon, "Everyday Functioning Benefits from an Assisted Living Platform amongst Frail Older Adults and Their Caregivers," *Frontiers in Aging Neuroscience*, vol. 9, pp. 1–12, Sep. 2017. [Online]. Available: https://hal.inria.fr/hal-01597680

[13] B. Bertran, J. Bruneau, D. Cassou, N. Loriant, E. Balland, and C. Consel, "DiaSuite: A tool suite to develop sense/compute/control applications," *Science of Computer Programming*, vol. 79, pp. 39–51, 2014.