Tool-based methodologies for developing assisted living services

Rafik Belloum

THÈSE

PRÉSENTÉE À

L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

par **Rafik Belloum**

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : INFORMATIQUE

Méthodologies outillées de développement de services dédiés à l'assistance domicilaire

---

Soutenue le 09 juillet 2020, devant le jury composé de :

Président du jury :	Laurent Billonnet,	Professeur à l'Université de Limoges
Directeur de thèse :	Charles Consel,	Professeur à Bordeaux INP
Rapporteurs :	Laurent Billonnet,	Professeur à l'Université de Limoges
	Jean-Michel Bruel,	Professeur à l'Université de Toulouse
Examinateurs :	Daniel Négru,	Maître de Conférence HDR, Bordeaux INP
	Loïc Caroux,	Maître de Conférence, Université de Toulouse
Invité :	Nic Volanschi,	Advanced research position à Inria Bordeaux

– 2020 –

RAFIK BELLOUM

TOOL-BASED METHODOLOGIES FOR DEVELOPING ASSISTED LIVING SERVICES

INRIA BORDEAUX SUD-OUEST, FRANCE

LaBRI
Unité Mixte de Recherche CNRS (UMR 5800)
351 cours de la Libération
33405 Talence Cedex
France

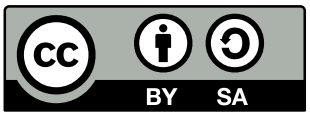
Équipe PHOENIX, INRIA Bordeaux Sud-Ouest
200 avenue de la Vieille Tour
33405 Talence Cedex
France

Université de Bordeaux

Inspired from the template provided by Paul van der Walt

WWW.DENKNERD.ORG

The typographic style of this document was inspired by Edward Tufte's book *Beautiful Evidence*, and typeset using LaTeX and a modified version of Kevin Godby's `tufte-book` class. The main text is typeset in *TeX Gyre Pagella*, which is based on Hermann Zapf's beautiful Palatino type face. The `typewriter` text is typeset in *Bera Mono*, originally developed by Bitstream, Inc.



Version: July 27, 2020.

Abstract

The growing population of older adults gives rise to a need for assistive computing systems that support independent living, to reduce the number of people being transferred to costly care facilities. The goal of assistive computing is to provide context-aware services that assist older adults in all aspects of daily life, for example, monitoring activities such as meal preparation and providing appointment or medication reminders. Despite much progress, the development of assistive services remains a challenge, because of a lack of supporting approaches and tools. This challenge involves: (1) coping with inter-individual variabilities (*e.g.*, home features and user routines and preferences) to deliver tailored services, (2) monitoring activities over long periods of time and (3) enabling care providers and/or professionals in aging to contribute their expert knowledge towards service development.

This dissertation presents several contributions to this topic. The primary contributions are two iterative methods dedicated to supporting the development of services that monitor activities of daily living (ADLs). Each of these methods is supported by a set of tools for collecting, analyzing and visualizing monitoring data. These tools ensure the agile development of accurate activity recognizers via a stepwise refinement of the analysis of sensor data. The first method, for recognizing ADLs, encompasses the main variations of a target activity by abstracting over descriptions reported by users. Beyond recognizing ADLs, the second method addresses long-term monitoring shortcomings (*e.g.*, sensor failures) and gives health professionals actionable insights into user activities. A final end-user approach is presented, which provides a tool to enable experts in aging to easily define assisted living services in smart homes.

The presented methodologies have been applied to an assisted living platform for aging in place, deployed in the home of 140 users. Experimental results show the effectiveness of all the proposed methods. First, the recognition methodology has achieved an accuracy of 80%, rising to 88% when considering the more routinized participants of the experiment. Second, the method for long-term monitoring of ADLs mostly produced the same interpretations as an expert in activity analysis, who manually analyzed the longitudinal sensor datasets. Finally, the findings reveal good usability of the end-user tool, which has been tested by occupational therapists.

Keywords: Smart home, Assistive computing, Assisted living services, End-user development, Activity monitoring

Résumé

L'accroissement du vieillissement de la population entraîne l'émergence de technologies informatiques pervasives au service de l'aide à domicile, afin de réduire le nombre de personnes transférées dans des établissements de soins coûteux. L'objectif de l'informatique d'assistance est de fournir des services adaptés au contexte qui aident les personnes âgées dans tous les aspects de la vie quotidienne, par exemple en surveillant des activités telles que la préparation des repas et en leur rappelant leurs rendez-vous ou leurs médicaments. Malgré de nombreux progrès, le développement des services d'assistance reste un défi, en raison du manque d'approches et d'outils de soutien au développement. Ce défi implique : (1) tenir compte des variations interindividuelles (*e.g.*, les caractéristiques du domicile et les habitudes et préférences des utilisateurs), (2) surveiller les activités sur de longues périodes et (3) permettre aux experts du vieillissement de personaliser les services d'assistance.

Cette thèse présente plusieurs contributions à ce sujet. Les principales contributions sont deux méthodes itératives dédiées au soutien du développement de services d'assistance. Chacune de ces méthodes est soutenue par un ensemble d'outils pour la collecte, l'analyse et la visualisation des données de suivi. Ces outils assurent le développement agile de détecteurs d'activité précis grâce à un affinement progressif de l'analyse des données des capteurs. La première méthode, pour la reconnaissance des activités, utilise les déclarations de routines rapportées par les utilisateurs. Au-delà de la reconnaissance des activités, la deuxième méthode s'attaque aux difficultés de la surveillance à long terme (*e.g.*, les défaillances des capteurs) et donne aux professionnels de la santé des indications utiles sur les activités des utilisateurs. Enfin, une approche pour l'utilisateur final est présentée, qui fournit un outil permettant aux experts du vieillissement de définir facilement les services d'assistance domiciliaire dans les maisons intelligentes.

Les méthodologies présentées ont été appliquées à une plateforme d'assistance, déployée au domicile de 140 utilisateurs. Les résultats expérimentaux montrent l'efficacité de toutes les méthodes proposées.

Mots clés : Maison intelligente, Assistance domiciliaire, Développement par l'utilisateur final, Détection d'activités

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my doctoral supervisor, Charles Consel, for providing invaluable guidance throughout this research. Without his high scientific standards, expertise and immense knowledge, this dissertation would not have been possible.

My sincere thanks goes to Laurent Billonnet, Jean-Michel Bruel, Daniel Négru and Loïc Caroux for accepting to participate on my dissertation committee as well as for their insightful comments. I also want to express my deep thanks to Nic Volanschi for his patience, valuable suggestions and creative thoughts, which I really appreciate. I also want to thank him for accepting to participate as a guest on my dissertation committee. It was a real pleasure working with you and I look forward for further collaborations.

I would also like to thank my fellow lab mates of the Phoenix team for providing a fun and stimulating working environment. I am especially grateful to Antoine, Julien and Quentin for being there for me when I needed it, for all the technical support they provide and their valuable advice. I am also grateful to my colleagues from the human-related sciences for sharing their knowledge with me to better understand the field from a multidisciplinary perspective. I would like to thank Adrien for sharing with me the template for this dissertation, which saved me a lot of time.

I am extremely grateful to Flora for providing comments on the draft version of my work and for showing a great deal of understanding and patience when I needed it the most. Without your support, encouragement and amazing care, I would not have finished this dissertation. I would also like to thank my son Naël who helps us to always look on the bright side of life thanks to his energy, his joy of life and his love.

Many thanks goes to my brother and sisters: Adel, Naouel and Assia for their encouragement and the amazing discussions via WhatsApp that cheered me up when I need it.

Last but not the least, I would like to thank my parents: Sakina Lemrabet and Hocine Belloum, whose love and guidance are with me in whatever I pursue in life. I am deeply grateful for the support they provided me through my entire life and for always believed in me, without their sacrifice this achievement would not be possible.

Résumé étendu

Le vieillissement de la population pose un vaste défi sociétal pour répondre aux besoins des personnes âgées et leur permettre de vivre de façon autonome. Pour relever ce défi, une approche prometteuse s'articule autour de l'informatique d'assistance et consiste à équiper le domicile des personnes âgées de technologies informatiques ubiquitaires et de services contextuels dédiés à la surveillance et à l'assistance de leurs activités quotidiennes. Dans cette approche, plusieurs domaines d'expertise sont impliqués dans le développement des services, allant des sciences humaines et aidants professionnels pour l'analyse des besoins, jusqu'à l'informatique pour le développement des services. En effet, l'adoption d'une approche interdisciplinaire, centrée sur l'humain et faisant appel à un éventail de compétences pour développer les services d'aide à l'autonomie des personnes âgées joue un rôle clé dans l'efficacité et l'acceptation de ces technologies. Bien que cette approche ait montré de nombreux avantages pour le développement des services d'assistance dans des environnements réels, il reste encore des défis à relever pour exploiter le potentiel de l'informatique d'assistance afin de répondre aux besoins des utilisateurs et de garantir la précision des services fournis. Examinons de plus près ces défis.

Assurer le développement de reconnaisseurs d'activités précis. Pour favoriser l'autonomie, il est primordial de surveiller les activités quotidiennes des personnes âgées, telles que la routine du coucher et la préparation des repas, car elles donnent une indication fiable sur la préservation de l'autonomie et permettent de prévenir les situations indésirables (*e.g.*, le manque d'activité pendant la journée). Ce suivi a notamment pour but d'évaluer l'évolution des activités quotidiennes dans le temps. D'une part, un manque d'activité soudain est une information précieuse pour un aidant ou un professionnel de la santé qui peut aboutir à une intervention rapide. D'autre part, une augmentation constante des absences est utile pour un soignant afin d'anticiper les mesures de compensation (*e.g.*, un service

de livraison de repas).

La surveillance de l'activité nécessite des méthodes de développement capables de fournir systématiquement des reconnaisseurs d'activités suffisamment précis pour être fiables et acceptés par les utilisateurs. En effet, compte tenu de leur imbrication dans la vie quotidienne des usagers, des détecteurs d'activités peu précis peuvent faire plus de mal que de bien. Par exemple, imaginons un détecteur d'activité qui reconnaît faussement des activités et envoie des rappels erronés à un utilisateur fragile et à son aidant. Au mieux, un tel service serait rapidement ignoré et/ou débranché par l'utilisateur ; au pire, il aurait un effet délétère sur lui.

Le principal défi à relever lors de l'implémentation de services de reconnaissance d'activités est de les rendre à la fois génériques et spécifiques : génériques pour faire face à un large éventail de configurations domestiques et de routines d'utilisateurs, et spécifiques pour détecter les activités avec un niveau de précision suffisant. Les approches basées sur l'apprentissage automatique sont très puissantes pour traiter un volume potentiellement élevé de données et fournir des réponses statistiquement correctes. Cependant, ces réponses peuvent ne pas être suffisamment prévisibles, ni faciles à expliquer aux personnes âgées et aux aidants. De plus, les approches basées sur l'apprentissage automatique nécessitent une grande quantité de données d'apprentissage (parfois étiquetées par des experts) afin d'être efficacement spécialisées pour chaque configuration domicile/utilisateur. Ces limitations suggèrent que les solutions déterministes de reconnaissance d'activités sont mieux adaptées à l'informatique d'assistance et que les reconnaisseurs d'activités devraient être à la fois génériques pour passer à l'échelle et personnalisables pour tenir compte des spécificités de l'utilisateur/du domicile.

Assurer le suivi des activités sur le long terme. La surveillance des activités quotidiennes dans un environnement réel, sur une longue période est très difficile. En plus de rendre les systèmes de surveillance des activités personnalisables pour tenir compte des variations interindividuelles, ces systèmes doivent relever d'autres défis importants.

Les systèmes de surveillance d'activités doivent tenir compte des pannes des capteurs. L'importance de tenir compte des défaillances des capteurs découle principalement du fait qu'elles sont inhérentes au déploiement à long terme dans des environnements réels de maisons intelligentes. Mais la surveillance des pannes est également essentielle pour interpréter les résultats des détecteurs d'activités. En effet, en l'absence d'activité

détectée pendant une certaine période, il est essentiel de distinguer les cas où l'activité n'a pas été réalisée et ceux où les capteurs ont été dysfonctionnels, afin de surveiller de manière fiable l'état fonctionnel des personnes âgées. Bien que prometteuse, la recherche sur les caractéristiques de la défaillance des capteurs est encore étudiée dans un cadre autre que celui d'un environnement réel : un laboratoire dédié aux études expérimentales.

Les informations sur l'activité pourraient être analysées pour évaluer le changement ou l'évolution de la routine des utilisateurs, notamment pour détecter les premiers signes de déclin cognitif (comme une diminution du temps de sommeil). Plus précisément, ces informations pourraient être utilisées par les professionnels de la santé dans le domaine du vieillissement, tels que les ergothérapeutes et les gériatres, pour déterminer l'assistance nécessaire. Dans le cadre d'études longitudinales, une quantité importante de données de capteurs contenant des informations sur les activités des utilisateurs est générée. Ainsi, pour interpréter cette masse de données, il est important de proposer des approches et des outils capables de visualiser les activités des utilisateurs de manière synoptique.

Permettre le développement de services par l'utilisateur final. Les aidants sont les mieux placés pour évaluer les besoins spécifiques des personnes âgées en terme d'assistance nécessaire à une vie autonome, car ils observent quotidiennement leurs habitudes, leurs besoins et leurs préférences. En outre, les personnes âgées ont tendance à sous-estimer leurs difficultés quotidiennes, ce qui oblige leurs aidants à compléter l'analyse des besoins. Ainsi, un facteur clé pour fournir une assistance personnalisable est de tirer parti des connaissances et de l'expertise des aidants dans le développement des services. Le problème est que les aidants manquent souvent de compétences en programmation. De plus en plus, les chercheurs s'efforcent de fournir des outils qui soutiennent la programmation par des utilisateurs finaux. Cependant, la plupart de ces outils nécessitent une longue période de familiarisation. C'est pourquoi il est nécessaire de proposer des outils pour les aidants non-programmeurs qui permettent une facilité d'utilisation immédiate afin d'atteindre une évolutivité en terme de besoins soutenus.

Contributions

Cette thèse présente trois méthodologies outillées qui soutiennent et facilitent le développement de services d'assistance

précis. Chacune de ces méthodologies répond à un défi identifié ci-dessus, à savoir la reconnaissance précise des activités, le suivi à long terme de ces activités et la possibilité de développer des services par l'utilisateur final.

Une méthode outiliée de développement de reconnaisseurs d'activités. Nous présentons une approche systématique pour développer des services précis de reconnaissance d'activité, basée sur une méthode outillée. Pour atteindre la précision, notre méthode consiste en un processus de développement en plusieurs étapes qui fait abstraction des descriptions des activités clés de l'utilisateur pour couvrir les variabilités inter-individuelles, tout en assurant une personnalisation appropriée en ce qui concerne les spécificités de l'utilisateur. Cette méthode de développement est itérative et permet d'ajuster les paramètres d'un détecteur d'activité pour maximiser sa précision.

Une méthodologie outiliée pour le suivi des activités sur le long terme. Nous proposons une méthode outillée pour la surveillance à long terme des activités des personnes âgées. Cette méthodologie couvre les étapes clés de la définition d'un processus de surveillance de ces activités. Ces étapes sont décrites de manière uniforme avec des règles concises et de haut niveau pour détecter les pannes des capteurs ou les activités. En outre, pour permettre aux soignants de surveiller le déclin fonctionnel des personnes âgées et de déterminer l'assistance nécessaire, notre méthodologie comprend un outil de visualisation, dédié à la gestion longitudinale des activités des utilisateurs. Nous avons mené une étude préliminaire [1] pour évaluer la fiabilité intra- et inter-participants de notre méthodologie, en utilisant des ensembles de données longitudinales, collectées sur plusieurs mois.

[1] La présente étude de cas est accessible à l'URL suivante : https://gitlab.inria.fr/rbelloum/reproducibilitymonitoring.git

Méthode de développement des services d'assistance par l'aidant. Afin de fournir une assistance personnalisée aux personnes âgées, nous présentons une méthode de développement des services d'assistance par l'utilisateur final. Cette approche comprend deux étapes : (1) une taxonomie des activités pour guider l'aidant dans la définition des services d'assistance ; (2) un wizard, qui permet à l'aidant d'exprimer facilement un service. Notre approche a été implémentée. Notre wizard a été utilisé avec succès pour définir les services existants programmés manuellement et développés avec un langage de programmation généraliste (Java). Les services résultants ont été exécutés par une vraie plateforme d'assistance et déployés au domicile de nos participants.

Organisation du manuscrit

Ce document est organisé comme suit :

Chapitre 2 traite des travaux connexes, couvrant les points saillants de nos méthodologies. Tout d'abord, nous discutons des caractéristiques et des exigences qu'implique le suivi des activités des personnes âgées. Ensuite, nous passons en revue les approches existantes pour reconnaître ces activités. Enfin, nous examinons certaines approches informatiques existantes qui soutiennent le développement d'applications contextuelles.

Chapitre 3 présente une méthode agile pour développer des détecteurs d'activités précis qui couvrent les variabilités inter-individuelles des utilisateurs. Pour illustrer notre approche, nous avons implémenté 6 détecteurs d'activités au domicile de 5 personnes âgées. Pour évaluer la précision de nos services, leurs résultats ont été comparés aux activités déclarées par nos participants sur une période de 5 jours. Cette expérience montre que 80% des résultats de nos détecteurs d'activités ont été confirmés par les utilisateurs, et 88% si l'on considère les quatre participants les plus routinisés.

Chapitre 4 présente une méthode disciplinée et reproductible de surveillance longitudinale des activités humaines. Par rapport à la méthodologie de Chapitre 3, nous montrons comment cette approche contribue (1) à améliorer la reproductibilité et (2) à raccourcir le cycle de développement des détecteurs d'activités grâce à des règles de surveillance de haut niveau et concises. Pour valider notre approche, nous présentons un ensemble de règles dédiées à la surveillance des activités des personnes âgées dans leurs domiciles. En utilisant la théorie de la détection du signal, nous avons montré que nos règles produisaient les mêmes interprétations qu'un expert en analyse d'activités, qui a analysé manuellement 5 ensembles de données de capteurs issues d'environnements réels.

Chapitre 5 présente une approche complète de développement de services de soutien à l'activité par les utilisateurs finaux. Par rapport aux méthodologies précédentes des Chapitres 3 et 4, nous élevons encore le niveau d'abstraction auquel les reconnaisseurs d'activité sont développés, afin de rendre leur personnalisation accessible aux utilisateurs finaux. Pour ce faire, nous présentons un wizard qui permet aux experts du vieillissement de définir des services d'assistance à partir d'une

taxonomie d'activités cibles, et nous montrons comment le wizard est interfacé avec une plateforme d'assistance. Nous évaluons la facilité d'utilisation de notre outil avec 5 professionnels du vieillissement (ergothérapeutes).

Chapitre 6 détaille les conclusions de cette thèse et discute des pistes de recherche en cours et à venir.

Contents

List of Figures

List of Tables

Dedicated to my grandmother.

1
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 context-aware services dedicated to monitoring and assisting their daily activities [1]. In this approach, several areas of expertise are involved in service development, ranging from human-related science and caregivers, to addressing needs analysis, to computer science for software development. In fact, following an interdisciplinary, human-centered approach that involves a range of expertise to develop assisted living services [2] for older adults plays a key role towards achieving effectiveness and acceptance of these technologies [3]. Although this approach has shown numerous benefits for assistive service development in real-world deployment, there are still challenges that need to be addressed to harness the potential of assistive computing towards meeting users' needs and ensuring the accuracy of the services delivered.

[1] Charles Consel [2018]. "Assistive computing: a human-centered approach to developing computing support for cognition." In: *2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, pp. 23–32.

[2] Assistive services take the form of applications that leverage the capabilities of the underlying infrastructure, such as sensing, actuating, user interaction, and networking. Examples of such services include agenda reminders, collaborative games, sleep monitor, and pedometer.

[3] Lucile Dupuy et al. [2017]. "Everyday Functioning Benefits from an Assisted Living Platform amongst Frail Older Adults and Their Caregivers." In: *Frontiers in aging neuroscience* 9, p. 302.

Contents

Overview

- Overview of the challenges of developing services that monitor home-based activities.
- Overview of the main research contributions presented in this dissertation.

1.1 *Accurate Activity Recognizers*

To support independent living, 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 [4]. On the one hand, a sudden surge in activity misses is a 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.,* meal delivery service).

[4] Barnan Das et al. [October 2012]. "PUCK: an automated prompting system for smart environments: toward achieving automated prompting—challenges involved." en. In: *Personal and Ubiquitous Computing* 16.7, pp. 859–873.

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 would be quickly ignored and/or unplugged by the user; at worse, it would have a deleterious effect on them [5].

[5] A.J. Bernheim Brush et al. [2011]. "Home automation in the wild: challenges and opportunities." en. In: *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*. Vancouver, BC, Canada: ACM Press, p. 2115.

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 potentially a 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.

1.2 *Long-Term Monitoring*

Monitoring daily activities in the wild (*i.e.,* real-life setting) over a long period of time is very challenging. In addition

to make activity monitoring systems customizable to account for inter-individual variations, these systems need to address other key challenges.

Addressing sensor failures. In a real-life setting, monitoring systems have to account for sensor failure. The importance of considering sensor failure stems primarily from the fact that it is inherent to long-term deployment in real smart home environments. But monitoring sensor failure is also essential for interpreting the results of activity recognizers. In fact, in the absence of an activity detected during a certain period, it is essential to distinguish between cases in which the activity has not been carried out and those in which the sensors have been dysfunctional, to reliably monitor the functional status of older adults. Although promising, research into the characteristics of sensor failure is still being studied in a non-real-life setting: a home dedicated to experimental studies [6].

Long-term analysis of activity data. Activity information could be analyzed to assess change or evolution of user routine, especially to detect early signs of cognitive decline (such as a decrease in sleep time) [7]. Specifically, such information could be used by health professionals in aging, such as occupational therapists and geriatricians, to determine what assisted support is needed. In the context of longitudinal studies, a sizeable amount of sensor data containing information on user activities is generated. Thus, to interpret this mass of data, it is important to propose approaches and tools capable of visualizing user activities in a synoptic way.

[6] Nancy ElHady and Julien Provost [June 2018]. "A Systematic Survey on Sensor Failure Detection and Fault-Tolerance in Ambient Assisted Living." en. In: *Sensors* 18.7, p. 1991.

[7] Loïc Caroux, Charles Consel, Lucile Dupuy, and Hélène Sauzeon [2018]. "Towards context-aware assistive applications for aging in place via real-life-proof activity detection." In: *Journal of ambient intelligence and smart environments* 10.6, pp. 445–459.

1.3 *Enabling End-User Development of Services*

Caregivers are best suited to assess the specific needs of older adults in terms of the assistance required for independent living, as they observe daily their habits, needs and preferences. Furthermore, older adults tend to under estimate their daily difficulties, requiring their caregivers to complement the needs analysis [8]. Thus, a key factor in providing customizable assistance is to leverage the knowledge and expertise of caregivers in the development of services. However, caregivers often lack computer skills. Increasingly, researchers aim to provide tools that support end-user programming. However, most of these tools require a long period of familiarization. For this reason, it is required to propose tools for non-programmers that allow for immediate ease of use in order to achieve scalability in terms of supported needs.

[8] David A Gold [2012]. "An examination of instrumental activities of daily living assessment in older adults and mild cognitive impairment." In: *Journal of clinical and experimental neuropsychology* 34.1, pp. 11–34.

1.4 Main Contributions

This dissertation presents three tool-based methodologies that support and facilitate the development of accurate assistive services. Each of these methodologies addresses a challenge identified above, namely, accurate activity recognizers, long-term monitoring, and enabling end-user development of services.

A tooled method to develop activity recognizers. We present a systematic approach to developing accurate activity recognizers, based on a tooled method. To achieve accuracy, our method consists of a multi-step development process that abstracts over the user descriptions of their key activities to cover 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. We validate our tool-based method by measuring the accuracy of our set of activity recognizers in a case study.

A tooled method for long-term monitoring. We propose a tool-based methodology for long-term activity monitoring of older adults. 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 for detecting sensor failures or ADLs. 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. We have conducted a preliminary study [9] to evaluate the intra- and inter-participant consistency of our methodology, using longitudinal datasets, collected over several months.

End-user method to develop assistive services. To provide a personalized assistance to older adults, we present an end-user approach for developing assistive services. This approach consist of two stages: (1) a taxonomy of home activities to guide the caregiver in defining assisted living services; (2) a wizard, which allows a caregiver to easily and practically express a service. Our approach has been implemented. Our wizard has been successfully used to define existing manually-programmed [10], activity-supporting services. The resulting services have been deployed and executed by an existing assisted living platform deployed in the home of community-dwelling individuals.

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

[10] The services developed with a mainstream programming language (*i.e.,* Java).

1.5 Outline

The reminder of this dissertation is organised as follows:

Chapter 2 discusses the related work, covering the salient features of our methodologies. First, we discuss the characteristics and requirements entailed by the monitoring of older adults' activities. Then, we review the existing approaches to recognize activities. Finally, we investigate some existing computing approaches that support the development of context-aware applications.

Chapter 3 presents a disciplined method to develop accurate activity recognizers that cover inter-individual variabilities of users. Using this approach, we implemented 6 activity recognizers in the home of 5 older adults. To assess the accuracy of our service 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, rising to 88% when considering the four more routinized participants.

Chapter 4 presents a a disciplined and reproducible method for longitudinal monitoring of human activities. Compared to the methodology of Chapter 3, we show how this approach contributes to (1) improving reproducibility and (2) shorten the development cycle of activity detectors via high-level and concise monitoring rules. Using the signal detection theory, we have shown that our rules produced the same interpretations as an expert in activity analysis, who manually analyzed 5 sensor datasets from real-life settings.

Chapter 5 presents a complete approach to developing activity-supporting services by end users. Compared to the previous methodologies of Chapters 3 and 4, we further raise the level of abstraction at which activity recognizers are developed, in order to make their customization accessible to end-users. To do so, we describe an end-user tool (*i.e.,* wizard) that allows caregivers to define assistive services within a taxonomy of the target activities, and we show how the wizard is interfaced with a smart home platform. We assess the usability of our wizard with 5 professionals in aging (*i.e.,* occupational therapists).

Chapter 6 details the conclusions of this dissertation and discusses the ongoing and future research avenues.

2
Related Work

The are many possible services that assistive computing systems can offer to older adults for prolonging aging in place within a smart home. In order to provide these services, computing approaches and tools are required to support the development of services that monitor home-based activities. In this chapter, we investigate smart homes that provide an infrastructure for services monitoring daily activities. We then examine these activities, drawing from how they are modeled and classified in the literature, we also investigate their characteristics in the context of our target population, namely older adults, and the impact of aging on them. Next, we review previous studies of sensor-based activity monitoring, examining what types of sensors and experimental settings were used. Finally, we review existing computing approaches to monitoring ADLs and the existing works aiming to simplify and support the development of context-aware services in smart homes. This review is done with respect to the key challenges identified in Chapter 1.

Contents

Overview

- A review of key characteristics and requirements involved in the activity monitoring of older adults in smart homes.
- A review of existing programming support and abstractions dedicated to the development of assisted living services.

2.1 Smart Homes

A smart home is commonly viewed as a set of connected objects, allowing the various home components to be controlled (heating, shutters, garage door, entrance gate, electrical outlets, *etc.*), and providing technical solutions to meet comfort needs (energy management, optimization of lighting and heating), security (alarm) and communication (remote controls, visual or audible signals, *etc.*) [Aldrich 2003]. Despite the many benefits of smart home automation, it does not address the specific needs of individuals with cognitive decline and/or disability. Their needs consist of environmental support to helping them perform their daily activities, as increasingly evidenced [Morrow and Rogers 2008; Reijnders et al. 2013]. As such, these needs go beyond existing forms of home automation but could leverage this infrastructure towards forming an environmental support. Research on applying smart homes to assisted living is a young field [Rashidi and Mihailidis 2013].

In recent years, researchers have been developing 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 [Kaye et al. 2011], 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 [Consel et al. 2017], 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).

2.2 Older-Adult Daily Activities

Classification of ADLs. The autonomous performance of ADLs is an important factor to promote independence in everyday activities [Fisk et al. 2018]. There is an extensive literature on activities of daily living, produced by such disciplines as occupational therapy (*e.g.,* [Townsend and Polatajko 2007]), human factors (*e.g.,* [Czaja et al. 1993]), psychology (*e.g.,* [Ormel et al. 2002]). ADLs are generally divided into two categories: basic activities (BADLs) that are necessary for fundamental functioning – eating, getting dressed, looking after the appearance, *etc.*– and instrumental ADLs (IADLs) that are necessary for independent living – cleaning and maintaining the house, prepar-

ing meals, shopping for groceries and necessities, leisure, taking medications, *etc.* [Lawton and Brody 1969]. The disciplines producing the classification of activities pursue various goals, ranging from evaluating the functional status of an individual, to devising an occupational rehabilitation program. Our goal is complementary in that we aim to develop a taxonomy of home activities, which serves as a framework for caregivers to define technology-based assistive services. The aim of this framework is to guide the caregivers in a step-by-step process in identifying and declaring the specificities of the user needs. Refining this process should contribute to develop a tool that supports service development. This opportunity is explored in Chapter 5 in order to address the challenge presented in Chapter 1, Section 1.3.

Impact of aging. 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 [Hong and Nugent 2013]. 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 [Bergua et al. 2013]. 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 [Caroux, Consel, Dupuy, and Sauzéon 2014].

2.3 *Range of Sensors*

In the context of a smart home 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 [Logan et al. 2007]. Note that because of privacy and intrusiveness concerns, activity monitoring for older adults often precludes the use of cameras when studies are conducted in their homes [Hossain 2014].

2.4 Experimental Settings

Most available sensor data targeting activity recognition are recorded in a controlled environment dedicated to experimental studies [Logan et al. 2007; Seelye et al. 2013]. 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 [Caroux, Consel, Dupuy, and Sauzeon 2018].

2.5 Computing Support for ADLs

The research in computing support to monitor activities can be decomposed in two topics: 1) the activity recognition techniques, and 2) the detection of sensor failures, as well as user routine deviations.

Activity recognition. Research on sensor-based activity recognition has made significant progress and is attracting growing attention in a number of application domains, and in particular context-aware services. Approaches to activity recognition are mainly based on machine learning or driven by user knowledge [Dawadi, D. J. Cook, and Schmitter-Edgecombe 2013].

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 [Logan et al. 2007]. The cost of the labelling task and its sensitivity to changes over time and across individuals, which occur in real homes [Logan et al. 2007], 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 [Dawadi, D. J. Cook, and Schmitter-Edgecombe 2013; Dawadi, D. J. Cook, Schmitter-Edgecombe, and Parsey 2013].

As introduced earlier, a knowledge-driven approach relies on routine declarations of users in their home to create activity models [Caroux, Consel, Dupuy, and Sauzeon 2018]. 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. This approach 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) [Consel et al. 2017] Although, activity verification has shown promising results, developing activity recognizers involves ad hoc and manual steps that prompt a need for methodological and tool support. 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 [Yordanova et al. 2019]. 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.

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 [Van Kasteren et al. 2010], Casas [D. Cook et al. 2009], Placelab [Logan et al. 2007]). Although these datasets include labelled activities for activity detection purposes, none of them include any labeling of data produced by faulty sensors [ElHady and Provost 2018]. 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 towards 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 [ElHady and Provost 2018].

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 [Hong and Nugent 2013; Riboni et al. 2011; Sarkar et al. 2010]. 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 ambient assisted living (AAL) and have been studied by Tran *et al.* [Tran et al. 2010], 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.

2.6 *Service Development*

To develop context-aware applications, approaches either use general programming languages (GPLs), domain-specific languages (DSLs) or end-user programming. This section investigates the different approaches to support the development of such applications in smart homes and examines their drawbacks.

Domain-specific languages

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 tedious to develop and evolve, making it difficult for researchers to build on each other's work via *reproducible research*. This issue even concerns DSLs for complex event processing (CEP), whose syntax and semantics can quickly obfuscate the detection logic [Volanschi, Carteron, et al. 2018]. A prerequisite to reproducibility is that data processing algorithms be accessible to and comprehensible for other researchers. These algorithms should be written in a *dedicated language*, which addresses the mentioned shortcomings of DSLs and GPLs. This opportunity is explored in Chapter 4.

End-user development

Beyond programming languages and domain-specific languages, end-user development (EUD) provides users with textual/visual forms of programming, which require little, if any, technical skills. However, even a successful end-user programming language, such as Scratch [Resnick et al. 2009], has a long learning curve for less tech-savvy users; they require user practice and time, which represent barriers for novices [Sutcliffe 2005].

In recent years, the field of smart home (SH) applications has been a major area of research in the context of the EUD. Early work on EUD for smart homes included iCAP, CAMP and MAPS. Dey *et al.* have shown through a user study that 95% of the smart services envisioned by users can be expressed as simple if-then rules [Dey et al. 2006]. Based on this finding, they have implemented a system called iCAP, which includes

a user interface for writing SH-related if-then rules, mixing icons and text. Rules are only allowed for rudimentary temporal relations (*e.g.,* sequences and durations). Another approach, pursued by Truong *et al.* allowed end-users to specify smart-home services by freely combining a small set of English words, relevant to a very narrow sub-domain, namely sound/video recording/playback applications [Truong et al. 2004]. In this approach, the services were specified using a system called CAMP, which automatically translated them into executable form. Truong *et al.* recognized that their approach was not suitable for broader service areas. Carmien *et al.* developed a system named MAPS that allows caregivers to define interactive prompting services for users with cognitive disabilities, using a film scripting interface [Carmien and Fischer 2008]. The services were not context aware and were limited to prompting (e.g., no activity monitoring).

An end-user approach to customizing smart homes is trigger-action programming (TAP), such as if-this-then-that, as pioneered by iCap and popularized by the website IFTTT and a variant such as AppsGate. They are prime examples of end-user development [Coutaz and Crowley 2016], 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 EUD 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 [Huang and Cakmak 2015].

As observed by Greenhalgh *et al.*, successful assistive service for older adults is often characterized by pragmatic customization, often performed by their caregivers [Greenhalgh et al. 2013]. However, customizing and developing asstive service can be quite an impediment for caregivers because it requires programming skills. Brich *et al.* argue that involving end-users in the development of services requires interfaces that need to be easy to understand and use, especially for less tech-savvy users [Brich et al. 2017], as are caregivers. A study of visual languages for smart spaces is reported by Reisinger *et al.*, where form-filling and data-flow programming are compared [Reisinger et al. 2017]. Form-filling allows participants to complete programming tasks faster and higher overall completion rate, whereas significantly more items are remembered when participants are being presented with a data-flow visualization. The authors recommend to blend both approaches for end-user programming of untrained users. Leveraging this

work, we are envisioning a wizard-based approach for end-user programming of services. This approach is presented in Chapter 5.

Chapter 2: Summary

Although promising, in the context of smart homes, most of existing approaches to monitor ADLs are still being studied in experimental settings. Furthermore, the computing techniques to support ADLs (*e.g.,* machine learning algorithms) often used in this context, face major challenges when applied to computing systems supporting individuals with cognitive decline and/or disability. Whether supervised or unsupervised, in a naturalistic setting, machine learning-based systems have to account for changes in sensors (*e.g.,* moved, broken, replaced) and changes in activities (*e.g.,* new activity patterns due to declining/acquired abilities). Putting these systems to practice still requires research.

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 towards making research on activity detection *reproducible*. Additionally, further research in EUD is needed to *allow non-experts* in programming to define *personalized* smart home services without relying on programming.

3
A Tooled Method for Developing Activity Recognizers

This chapter presents a disciplined and agile method dedicated to producing accurate and rapidly customizable activity recognizers that cover fine-grained specificities of users. This method is (1) knowledge-based in that it involves declarations from users and their caregivers to drive the service customization process, 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. [1].

[1] This work has been submitted: Rafik Belloum et al. [2020]. "A Tooled Method for Developing Knowledge-Based Activity Recognizers."

Contents

Contributions

- A multi-step development method that leverages user declarations and generalizes over inter-individual variabilities.
- A visualization tool that enables the rapid customization of generic activity recognizers.
- An experimental study that assesses the accuracy of our approach by matching the results of activity recognizers against the user truth.

3.1 Introduction

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 in Chapter 2. 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 [4] 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 [5]. 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 declared 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.

We take here 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 but generalizes over inter-individual variabilities while allowing proper customization with respect to user specificities. This development method is iterative and allows to adjust the parameters of an activity

[2] Loïc Caroux, Charles Consel, Lucile Dupuy, and Hélène Sauzéon [October 2014]. "Verification of Daily Activities of Older Adults: A Simple, Non-Intrusive, Low-Cost Approach." In: *ASSETS - The 16th International ACM SIGACCESS Conference on Computers and Accessibility*. Rochester, NY, United States, pp. 43–50.

[3] Loıc Caroux et al. [2018]. "Towards context-aware assistive applications for aging in place via real-life-proof activity detection." *In: Journal of ambient intelligence and smart environments 10.6,pp. 445–459.*

[4] Liming Chen et al. [June 2012]. "A Knowledge-Driven Approach to Activity Recognition in Smart Homes." In: *IEEE Transactions on Knowledge and Data Engineering* 24.6, pp. 961–974.

[5] Valérie Bergua et al. [2013]. "Restriction in instrumental activities of daily living in older persons: Association with preferences for routines and psychological vulnerability." In: *The International Journal of Aging and Human Development* 77.4, pp. 309–329.

recognizer to maximize its accuracy.

Our work makes the following key contributions. Firstly, our method is supported by a set of tools for collecting, analyzing and visualizing monitoring data. These tools ensure the agile development of generic activity recognizers and their rapid and effective customization to achieve accuracy. Secondly, we reveal the fact that user declarations have to be checked and ususally adjusted with respect to the real sensor data to ensure accurate activity recognition. Thirdly, we expand the range of target activities, compared to Caroux *et al*, and generalize their formula-based approach to cope with partially performed activities; a set of generic and customizable activity recognizers is presented. Finally, we validate our tool-based method by measuring the accuracy of our set of activity recognizers in a realistic case study on 5 users and 6 activities, namely, bedtime routine, wakeup routine, outings, preparation of breakfast, lunch and dinner.

3.2 *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 on average, 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 [7].

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 1 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 1 defines

[6] Charles Consel et al. [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, pp. 129–140

[7] Lucile Dupuy et al. [2017]. "Everyday Functioning Benefits from an Assisted Living Platform amongst Frail Older Adults and Their Caregivers." In: *Frontiers in aging neuroscience 9, p.302*

the function for each sensor deployed in a home, that is, the meaning of the sensor measurements.

Figure 1 displays the layout of an apartment fitted with the HomeAssist sensors, whose placements are guided by the declarations of its occupant.

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	Kitchen presence
Entrance	ContactS_E	Door open
	MotionD_E	Entrance presence
Bedroom	EMeter_L	Bedside lamp in use
	MotionD_B	Bedroom presence
Bathroom	MotionD_Ba	Bathroom presence
	MotionD_S	Shower/Bathtub presence
Toilet	MotionD_T	Toilet presence
Living room	MotionD_L	Living room presence

Table 1: HomeAssist sensors and their functions.

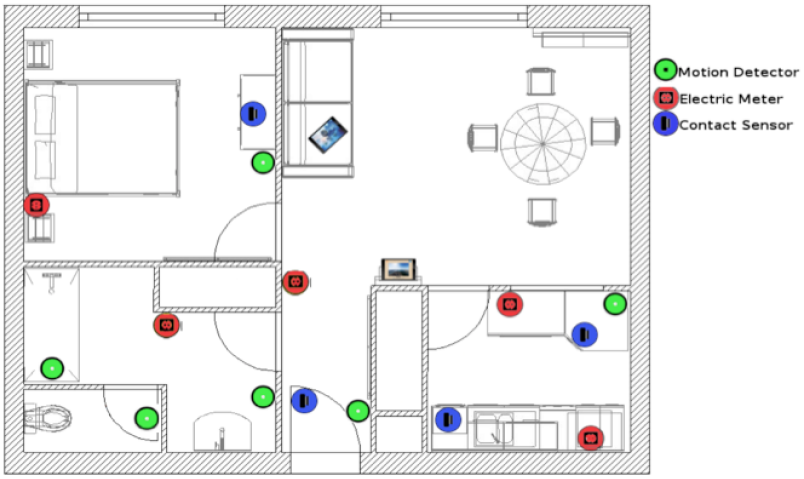


Figure 1: Example of an apartment layout with sensors.

3.3 *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 2. Let us examine the key concepts and steps, forming our approach.

Step 1 of our approach (noted "1. Declaration" in Figure 2) is the declaration of routines by the seniors and/or their caregivers (noted '(a)' in Figure 2). 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, wakeup time, transition time between bedroom and kitchen in the morning). These values serve as parameters for service customization (noted '(b)' in Figure 2).

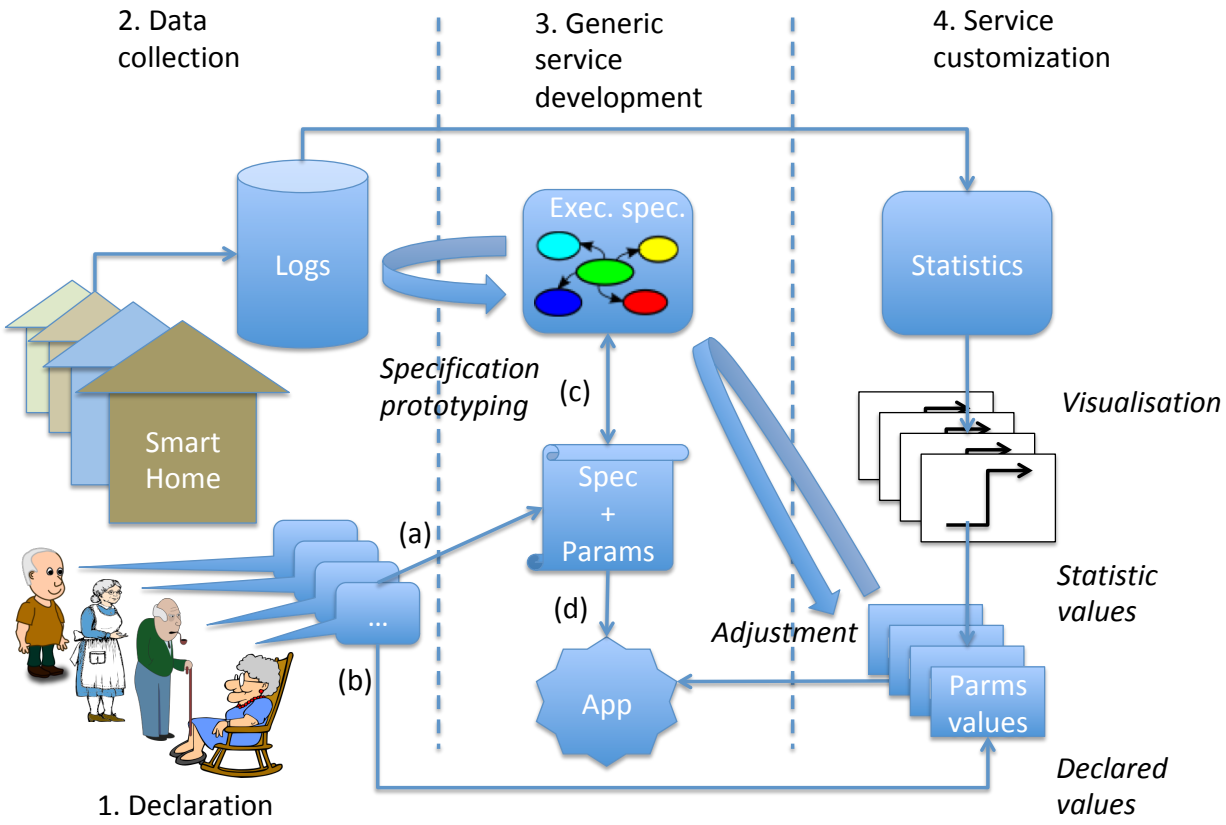


Figure 2: Overview of the development method.

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, 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. In our case, we use the Java programming language and the HomeAssist platform [8].

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 3.5, values provided by users are only estimates, and can rarely be used as final customization values. Thus, it is neces-

[8] Benjamin Bertran et al. [January 2014]. "DiaSuite: A tool suite to develop Sense/Compute/Control applications." en. In: *Science of Computer Programming* 79, pp.39–51.

sary to check these values against real logs, and perform the necessary adjustments. 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 visualization tool allows to execute that 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.

3.4 *Case Study*

We now present the case study 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.

Declaration and data collection

The declarations of activities of interest were gathered at the installation time of the platform in each home. In doing so, sensor logs started to be accumulated and provided a basis to assess the accuracy of user declarations. 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.

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

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.

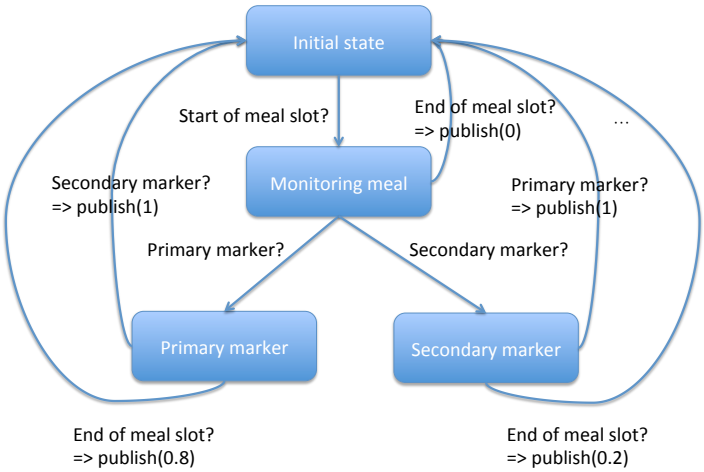


Figure 3: Automaton for recognizing the different meal routines.

Logic: The logic of this service is implemented by the automaton in Figure 3. 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).

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

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, while the cupboard door is always the secondary marker when detected open.

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.

Logic: The logic of this service is implemented by the *timed* automaton in Figure 4. 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 bed-

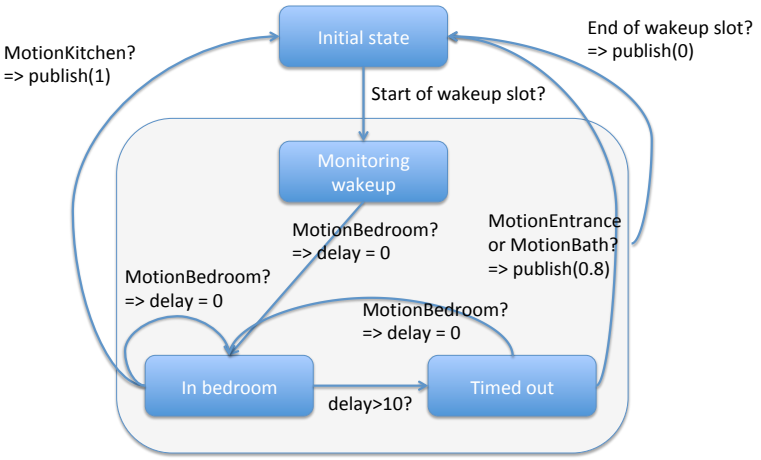


Figure 4: Timed automaton for recognizing the wakeup routine.

room". 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.

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

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 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.

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.

Logic: The logic of this service is implemented by the timed automaton in Figure 5. Starting in the initial state, a transition

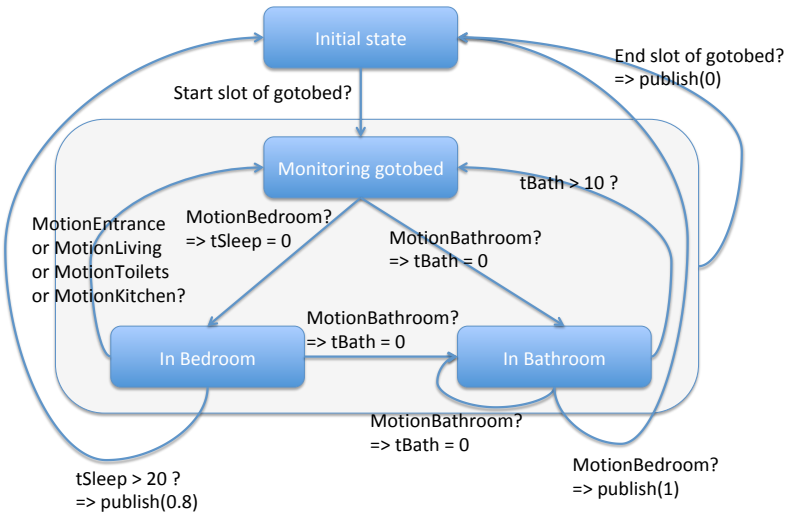


Figure 5: Timed automaton for recognizing the bedtime routine.

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).

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

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.

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.

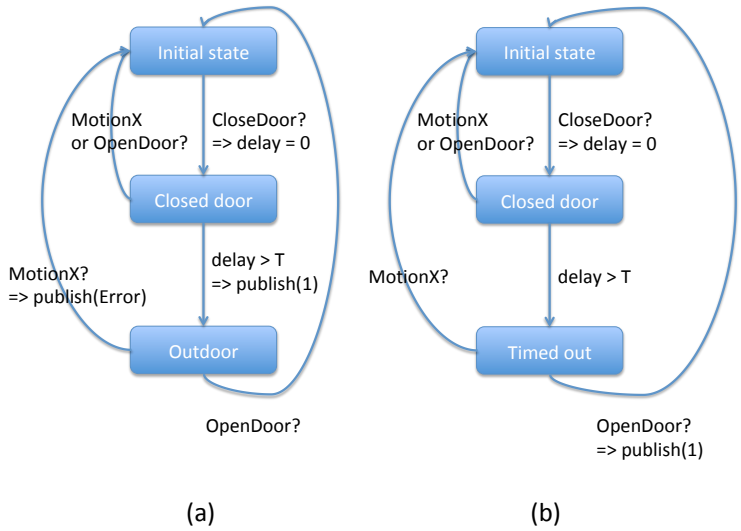


Figure 6: Timed automaton for recognizing outings: (a) in real time; (b) *a posteriori*.

Initial logic: The initial logic of the service was implemented by the timed automaton in Figure 6 (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.

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 6 (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.

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

Service customization

Once developed, the generic services can be customized, leveraging our visualization 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 7 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 con-

figuration parameters allow to make an assessment of the accuracy of the service recognizers, by executing the scripted service recognizers on the log. Table 2 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.

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

Table 2: Examples of service customization for one home-/user configuration.

Let us examine Table 2 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 7, 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 2): 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 days within the two-week 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 adjusted values of these are displayed in italics in Table 2. As can be seen, the time slots of *all* the rec-

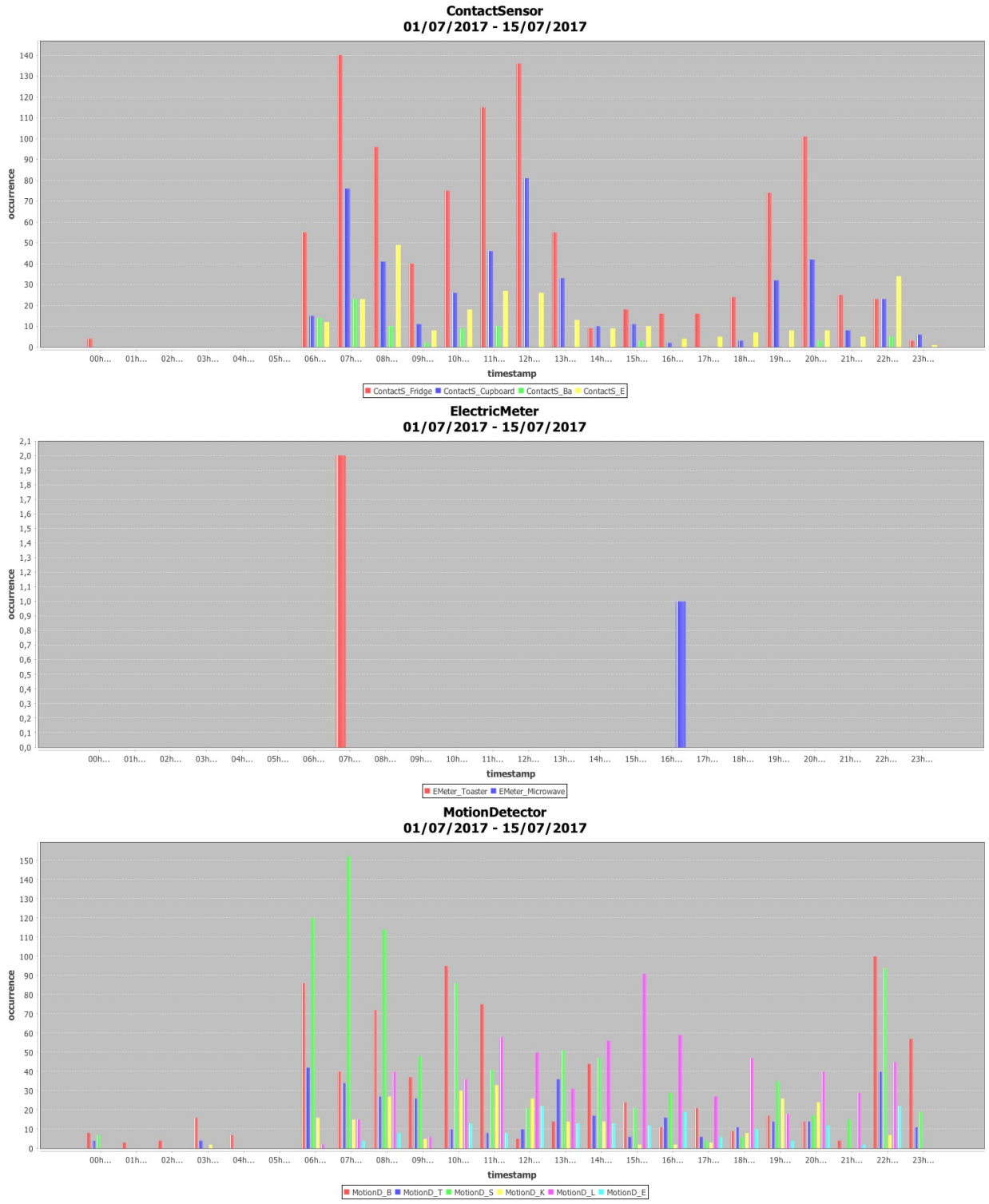


Figure 7: Histograms of sensors spread over a 24-hour period in a home: contact sensors (top), appliance uses (middle), motion sensors (bottom).

ognizers 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 2). By studying the histograms of the motion detectors (bottom part of Figure 7), 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%.

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 [9]. We then customized these services with respect to 5 different users and their home using the visualization tool, as described previously. 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 issued if activity is missed). This mode only logs the detected activities.

[9] Benjamin Bertran et al. [January 2014]. "DiaSuite: A tool suite to develop Sense/Compute/Control applications." en. In: *Science of Computer Programming* 79, pp. 39–51.

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.

3.5 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 recogniz-

ers, and more generally, of the services produced by our tooled method.

To achieve this user validation, 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.

The questionnaire data collected from our 5 participants are displayed in Table 3. Note that, because Participant B was bedridden during our study, the services to detect wakeup and outing were not installed. Note also that a technical difficulty prevented us from installing the outing detector in the home of Participant D [10]. Table 3 lists, for each day, participant, and activity, the score between 0 and 1 produced by the activity recognizer, and the corresponding user report (Yes or No). Recall that scores of 0 or 0.2 indicate a non-detected activity, while scores of 0.8 or 1 indicate a detected activity. The greyed cells in the table correspond to cases where the user did not agree with the activity detector.

[10] 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.

Activity	Day1		Day2		Day3		Day4		Day5	
	Service	User	Service	User	Service	User	Service	User	Service	User
Wake up	100	Yes	100	Yes	100	Yes	100	Yes	100	Yes
Breakfast	80	Yes	80	Yes	80	Yes	80	Yes	100	Yes
Lunch	0	No	100	Yes	100	Yes	100	Yes	100	Yes
Dinner	100	Yes	100	Yes	100	Yes	100	Yes	100	Yes
Go to bed	0	Yes	100	Yes	80	Yes	100	Yes	100	Yes
Exit	1	1	0	0	1	1	0	0	0	0
Wake up										
Breakfast	100	Yes	20	Yes	100	Yes	80	Yes	80	Yes
Lunch	100	Yes	100	Yes	100	Yes	100	Yes	100	Yes
Dinner	100	Yes	100	Yes	100	Yes	100	Yes	100	Yes
Go to bed	80	Yes	80	Yes	100	No	100	Yes	100	Yes
Exit										
Wake up	100	No	100	No	100	No	100	Yes	100	No
Breakfast	100	No	100	Yes	100	Yes	100	Yes	100	Yes
Lunch	100	Yes	20	Yes	0	Yes	0	Yes	20	Yes
Dinner	0	Yes	0	No	20	No	20	Yes	20	No
Go to bed	100	Yes	100	Yes	100	Yes	100	Yes	0	Yes
Exit	3	2~3	3	2~3	1	2~3	1	1	3	1
Wake up	80	Yes	0	Yes	80	Yes	80	Yes	0	No
Breakfast	100	Yes	100	Yes	100	Yes	100	Yes	100	No
Lunch	100	Yes	100	Yes	100	Yes	100	Yes	100	Yes
Dinner	20	Yes	0	Yes	100	Yes	100	Yes	100	Yes
Go to bed	80	Yes	80	Yes	0	Yes	0	Yes	80	Yes
Exit										
Wake up	100	Yes	100	Yes	100	Yes	100	Yes	100	Yes
Breakfast	100	Yes	100	No	100	Yes	100	Yes	100	Yes
Lunch	100	Yes	100	No	0	Yes	100	Yes	100	Yes
Dinner	100	Yes	100	Yes	0	Yes	100	Yes	80	Yes
Go to bed	80	Yes	80	Yes	100	Yes	80	Yes	80	Yes
Exit	0	0	0	0	1	1	0	0	1	1

Table 3: Comparison of activities detected by the services *vs.* reported by the users.

An overall view of the validity of our activity detectors is shown in Figure 8. The detailed counts of valid and invalid reports for each detector are given in Table 4.

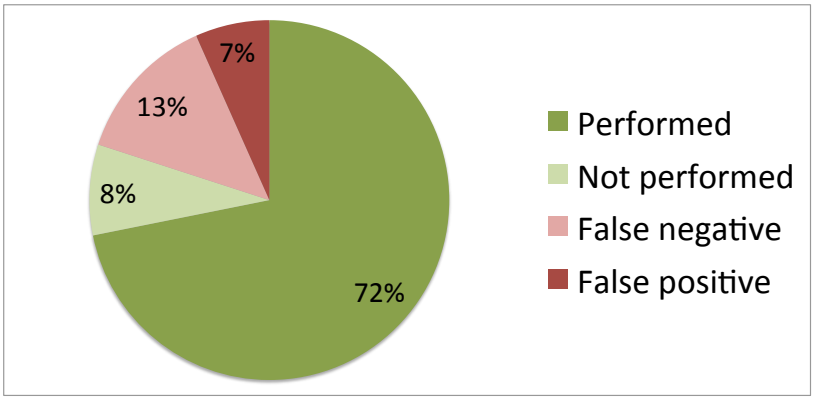


Figure 8: Overall validity of the detected activities.

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

Table 4: Detailed validity results for each activity detector.

3.6 *Discussion*

As shown by Figure 8, 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 4 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 Table 3 for these detectors, other interesting patterns are revealed. Namely, the wrong results pro-

duced by these two detectors are specific, with very few exceptions, to Participant C (a single exception for the wakeup detector, and only two 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. Consequently, obtaining an overall accuracy of 80% should provide older adults and their caregivers valuable information to support independent living.

Chapter 2: Summary

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 sensors 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. To assess the accuracy of these 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.

4
Long-Term Activity Monitoring

This chapter introduces a tool-based methodology that addresses the *long-term monitoring challenges* discussed in Chapter 1, Section 1.3. To pursue *reproducibility* and *shorten the development cycle*, this approach goes beyond mainstream programming languages used to monitor the activities in Chapter 3. In particular, this method uses a domain-specific language (DSL) to define concise, high-level monitoring rules. This language (1) allows an even more agile development than the tools used in the previous chapter 3 and (2) makes the monitoring algorithms comprehensible and accessible to other researchers [1].

[1] This work has been published: Rafik Belloum et al. [2020]. "A Tool-Based Methodology For Long-Term Activity Monitoring." In: *PETRA'20-Pervasive Technologies Related to Assistive Environments*.

Contents

Contributions

- An iterative process for developing concise, high-level monitoring rules that analyze sizeable sensor data to detect sensor failures or ADLs.
- A visualization tool to support the refinement of analysis rules during the iterative process and also to allow caregivers to monitor older adults longitudinally.
- An experimental study that uses Signal Detection Theory to validate the accuracy of monitoring rules.

4.1 Introduction

During the past few decades, steady progress has been made in developing sensor-based approaches, capable of recognizing a range of activities. However, monitoring activities in a real home and over a long period of time often defeats conceptual models developed in a controlled environment, as shown in Chapter 2. A methodology claiming to support the development of monitoring 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

We introduce 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, we present an iterative process that supports a stepwise refinement of the analysis of the sensor data. It 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 the activities of older adults longitudinally.

Using a dedicated language. To pursue reproducibility and allow analysis rules to evolve during the iterative process, our approach revolves around Allen, a domain-specific language dedicated to defining rules that analyze sensor data [2]. Specifically, we use this DSL 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 and they can be developed in a more agile manner than using a GPL or even a scripting language. As a byproduct, the use of this DSL makes the rules more comprehensible to other researchers, contributing to research reproducibility.

[2] Nic Volanschi, Bernard Serpette, et al. [December 2018]. "A Language for Online State Processing of Binary Sensors, Applied to Ambient Assisted Living." en. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2.4, pp. 1–26.

Putting the methodology into practice. We have applied our tool-based approach 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 our iterative process. The results have been validated, using signal detection theory, by comparing the results of our rules with a human observer. Our rules produced the same interpretation as the human judge, who manually analyzed the datasets.

4.2 *Methodology*

In this section, we present our 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.

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.

Note that our methodology is illustrated by sensor datasets from the HomeAssist project presented in Chapter 3, Section 3.2. The sensors used for our examples in this chapter are presented in Table 1 of Chapter 3.

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 5, 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 5, 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.

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

Table 5: Output of monitoring rules detecting activities and sensor failures

Let us now illustrate the first three steps of our approach by defining a rule that detects visits to the toilets. The fourth step is examined in Section 4.4, where a set of rules is evaluated using Signal Detection Theory and a human observer.

Writing a rule. Our 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 we leverage the HomeAssist project, we only consider 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`.

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 occurrence 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 4.3). 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.

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 5. 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 straightforward, 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 4.3) 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.

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 9. 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 9); and Situation 2 results in the absence of both bars (first two months in the figure).

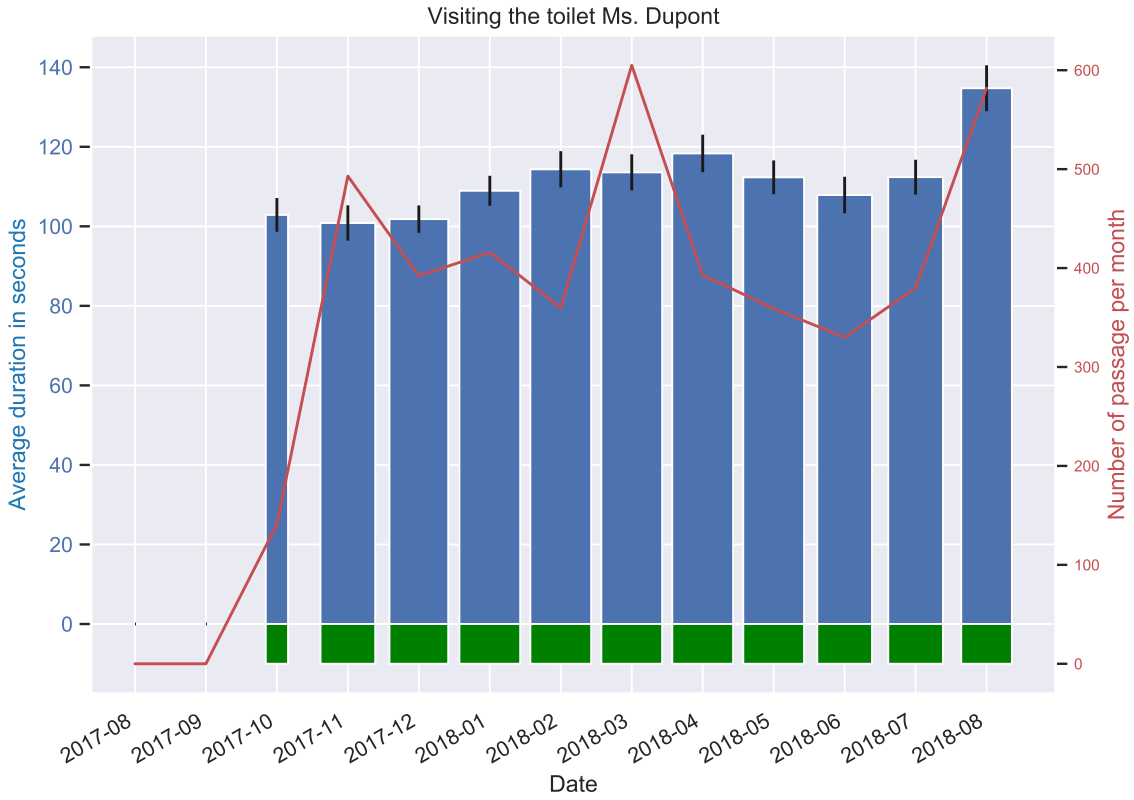


Figure 9: Visualization of toilet activity.

Example. Figure 9 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, we can see the periods during which the toilet detector properly worked in this user's home. During August and September 2017, we have 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.

4.3 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 chapter. The dataset of the other participants could not be disclosed for privacy reasons.

Outings

Let us first define the rules for each of the monitoring activities addressed by our case study. To do so, we introduce the notion of user-defined operator provided by Allen to allow 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.

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 our methodology, we manually analyzed samples of sensor data of participants. We 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, we added to the operator definition a minimal time of absence `T` before declaring an outing; it is introduced as 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).

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 at home (*i.e.*, `~outing`) and the contact sensor of the entrance door has not been activated during a given time `T`.

Visualization. Figure 10 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, our 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 our (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.

Sleeping

It would be unrealistic to aim at detecting actual sleep of an individual solely with ambient sensors. Instead, our goal is to detect when the user spends some quiet time during the night

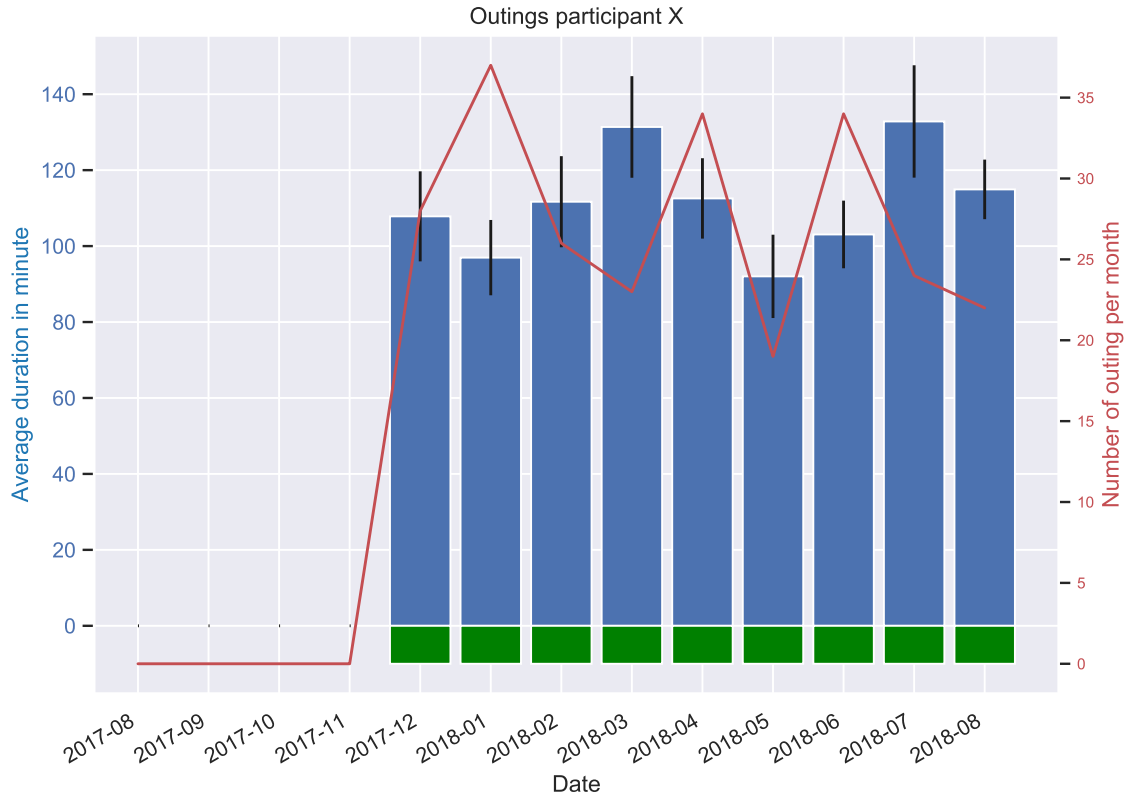


Figure 10: Visualization of outings.

in their bedroom. This activity should give an indication of how much sleep an individual is getting.

Activity detection. Let us incrementally define our notion of sleep. First, we assume that this activity occurs 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 4.2).

```
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.

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, we cover the two years 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, we introduced a new monitoring rule (definition omitted), which 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.

Failure detection for bedroom sensor. The same logic as that of toilet failure and door failure is used to detect failures of the bedroom motion detector. Therefore, we may reuse our user-defined operator.

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

Visualization. Figure 11 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 our (blue and) green bars, apart from October 2017 and February 2018 where the bars are thin.

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 the sub Section 4.3.

4.4 Evaluation of Activity Monitoring Rules

Using Signal Detection Theory [3] (SDT), an expert in activity

[3] Harold Stanislaw and Natasha Todorov [March 1999]. "Calculation of signal detection theory measures." en. In: *Behavior Research Methods, Instruments, & Computers* 31.1, pp. 137–149.

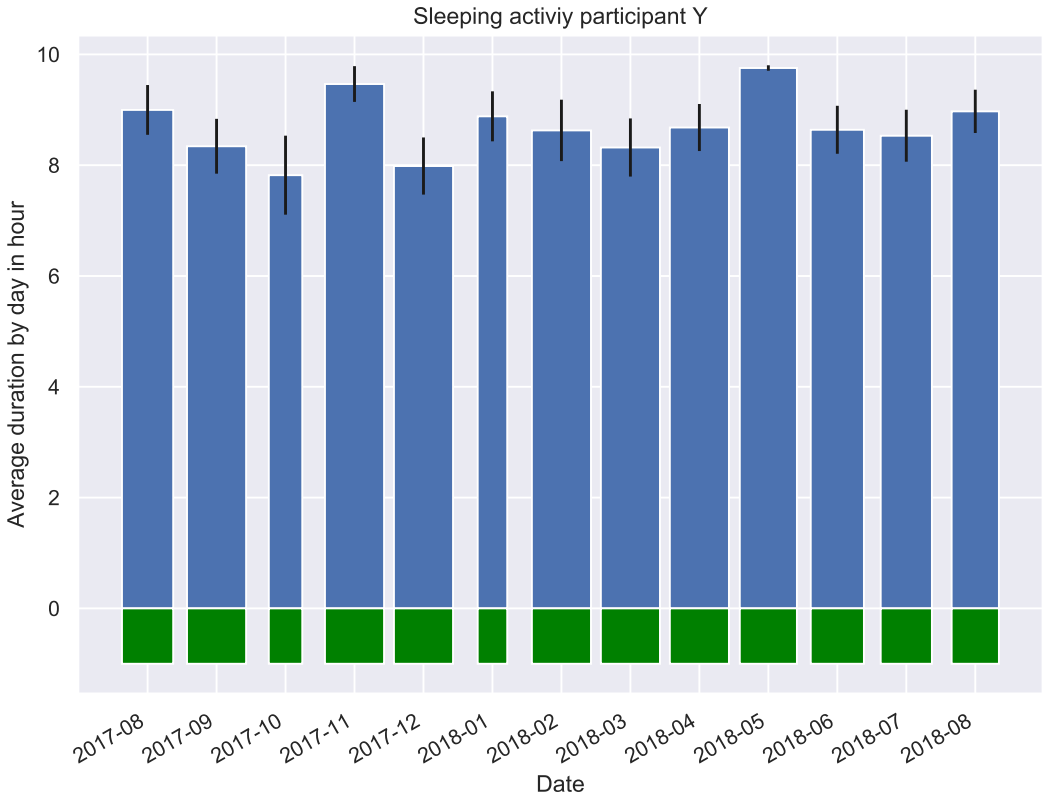


Figure 11: Visualization of sleeping activity.

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.

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, human judges 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.

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

Table 6: Evaluation of monitoring rules using SDT.

The complete results are shown in Table 6. 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.

Chapter 3: Summary

We have presented a tool-based methodology to develop knowledge-based rules dedicated to processing longitudinal, real-world, home-centric, sensor data. We have shown that our approach reliably detects older adults' activities and provides professional caregivers with actionable insights via a visualization tool.

Our work improves replicability of activity monitoring research in that it introduces an iterative process to develop concise and high-level activity-monitoring rules. Compared to the previous methodology of Chapter 3. we shorten the development cycle in the order of 1 hour-person instead of 1 person-day. Our 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. We illustrated our methodology 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, we have shown that our rules for detecting sensor failures and various activities (sleeping, toilet visits and outings) are accurate and reliable.

5
End-User Development of Activity-Supporting Services

As we saw in the previous Chapter 4, Allen allows the expression of high-level and comprehensible rules but this language is only accessible to programmers. In this chapter, we present how to leverage the knowledge and expertise of caregivers for service development, without relying on programming. To address this challenge of *end-user development* identified in Chapter 1, Section 1.3, we present here a tool-based approach that allows caregivers to define services. This approach consists of two main stages: 1) a wizard that allows caregivers to define an activity-supporting service,; 2) the wizard-generated service is uploaded in an existing smart home platform and interpreted by a dedicated component, carrying out the caregiver-defined service [1], [2].

[1] A preliminary version of this work has been published: Rafik Belloum [2020]. "End-user Development of Activity-Supporting Services for Smart Homes." In: *In Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom Workshops). Austin, Texas. 2020.*

[2] A full version of this work has been submitted: Rafik Belloum, Charles Consel, et al. [2020]. "Caregiver Development of Activity-Supporting Services for Smart Homes."

Contents

Contributions

- A Taxonomy for independent living to model the target activities to support by our methodology.
- A wizard that allows caregivers to define activity-supporting services without programming.
- A run time component, carrying out the caregiver-defined services to deliver assistance to users.
- An experimental study to evaluate the usability of the wizard by health professionals.

5.1 Introduction

We conducted such an experiment using HomeAssist platform, presented in Chapter 3, Section 3.2, to address the needs of three populations – older adults, persons with intellectual disability, and persons with autism spectrum disorders – in performing their home activities independently. Our aim was to explore assistive services with practical case studies, covering all aspects from collecting needs for specific users, to deploying services tailored to each user in their home. To do so, we formed an interdisciplinary group with a range of expertise, including psychology, occupational therapy, professional caregiving, and computer science. Early in the project, we found that, even though some services could apply to a range of users, those creating the most interest in users and caregivers often required specific interviews, analysis, and software development. Although specific, this work involved common stages that often was iterated: (1) analyzing needs with a user and/or their caregiver; (2) writing a specification of an assistive service with feedback from the user and/or their caregiver; (3) developing the assistive application for the smart home platform; (4) and, assessing the application with the user and/or their caregiver. Thanks to our human-centered approach, we were able to adjust the services during development, preventing any gap between the behavior of the services and the needs expressed by the user and/or their caregiver.

This systematic approach allowed us to produce services highly tailored to users, not necessarily knowing a priori the specific topics we would be addressing. Such services showed a positive impact on the users and their caregivers because they felt their individual needs had been taken into account and they were able to concretely assess the outcome of their inputs in the resulting services. As an illustration of this exploration, we developed a service to manage outside appointments of a user. This service uses a calendar to manage events. If an event is triggered and corresponds to an outside appointment, the service informs the user of the upcoming appointment early enough to give them time to prepare. Then, a sensor is used to check whether the user left home. If not, when the appointment is important (*e.g.,* health related), the caregiver is notified, in addition to the user. Other examples covered in this exploration included such activity areas as household chores (*i.e.,* homemaking), vocational, leisure and health management. Even though satisfying for users, our approach showed practical limitations, as we tried to grow the number of users and assistive goals: it did not scale up because of the

amount of time required to gather the user needs and develop the corresponding services.

To resolve our software development bottleneck, we set out to analyze the activity-supporting services we had developed. This analysis revealed that they had extensive common properties, which suggested that they formed a program family [3]. As reported in the literature, the variations and commonalities of a program family can be leveraged to factorize parts of the software development process. Factorization typically takes the form of domain-specific languages [4] and gives rise to program generation tools [5]. This finding was a key insight towards solving our software development bottleneck problem.

[3] David Lorge Parnas [1976]. "On the design and development of program families." In: *IEEE Transactions on software engineering* 1, pp. 1–9.

[4] Marjan Mernik et al. [2005]. "When and how to develop domain-specific languages." In: *ACM computing surveys (CSUR)* 37.4, pp. 316–344.

[5] Krzysztof Czarnecki et al. [2000]. *Generative programming: methods, tools, and applications*. Vol. 16. Addison Wesley Reading

Our approach

We propose a complete approach to developing activity-supporting services, ranging from the modeling of the target activities, to an end-user tool to define services, to a layer to run services in a smart home.

A taxonomy of activities. To model the activities to be supported by our approach, we have developed a taxonomy, drawing from 1) the taxonomies of home activities reported in the literature, 2) discussions with caregivers and 3) examining the assistive services that we developed during our experiment.

A wizard for activity-supporting services. To prevent any gap between the gathered user needs and the resulting activity-supporting service, we have developed a wizard, which allows a caregiver to express a service within our taxonomy of target activities. This wizard runs on a tablet and was designed and developed in close collaboration with three professionals in aging to ensure its usability. Specifically, at key development stages, we conducted interviews and usability tests.

Execution support for wizard-defined services. Our wizard generates a representation of a caregiver-defined service that is fed to the smart home platform. A dedicated layer is in charge of interpreting this representation to realize the service, leveraging available connected objects (*e.g.,* sensor), services (*e.g.,* calendar), and interaction modalities (*e.g.,* user notifications).

Validation. To validate our approach, (1) we used our wizard to define existing activity-supporting services and observed their behaviour equivalence when deployed in the home of our participants; (2) additionally, we used our wizard to define a number of new services to test the coverage of the target taxonomy of activities – the wizard-defined services targeted older adults, users with intellectual disability, and users with

autism; (3) finally, we conducted a study to measure the usability of the wizard with five occupational therapists (OTs) that used the wizard's features to respond to clinical fictitious situations; the results reveal a good usability of our wizard by OTs.

5.2 *Taxonomy of Activities for Independent Living*

First, we explore and organize activities of daily living and the needs they could address, according to users and their caregivers. Next, we present the assistive goals that result from activities of interest. Then, we examine the assistive applications that were (manually) developed, based on these assistive goals and deployed in the smart home of our participants. Leveraging these applications, we identify and analyze their commonalities and variabilities towards forming a taxonomy of technology-supported activities. This taxonomy will serve as a framework to define activity-supporting services.

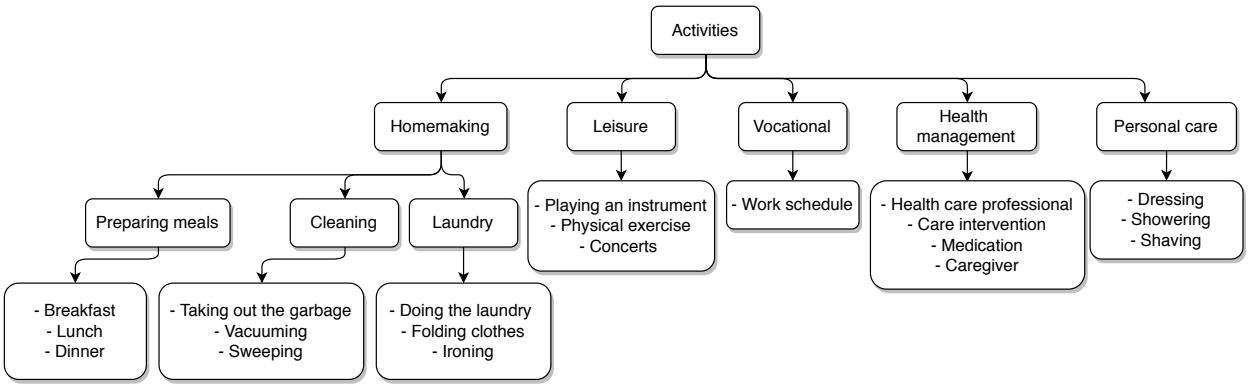


Figure 12: An extract of a taxonomy of home activities.

Exploring activities. To explore activities of daily living in the home, we leverage a taxonomy used by occupational therapists to assess the ability of individuals to live independently in their home [6]. An extract of this taxonomy is presented in Figure 12. As can be noticed, activities of daily living in the home are decomposed hierarchically: from general categories of activities at the root, to refined specific categories towards the bottom. Leaves define a set of concrete activities, such as taking out the garbage, vacuuming and sweeping. As such, this taxonomy allowed us to explore user needs in a systematic manner, down to specific activities, which can become assistive goals.

Assistive goals. Once an activity is targeted by a user and/or their caregiver as an assistive goal, we start specifying what

[6] Richard Bernard Dever [1988]. *Community living skills: A taxonomy.* American association on mental retardation

Activity	Assistive goal
Household chores – *Preparing meals (breakfast, lunch, dinner)*	Supervising and assisting meal preparation.
Sleep hygiene –*Bed times*	Supervising the user and suggesting bed times.
Vocational – *Going to work*	Supervising the user to ensure they leave home for work on time.
Household chores – *Vacuuming, sweeping, ironing, taking out the garbage, etc.*	Supervising and assisting household chores to ensure a well-managed home.
Leisure – *Swimming pool*	Supervising the user to ensure they leave home for swimming activity on time.
Leisure – *Practicing a musical instrument*	Supervising the user to ensure they practice their musical instrument.
Health management – *Healthcare visit*	Supervising the user to ensure they are ready for a visit of a healthcare professional.
Health management – *Medication*	Supervising the user to ensure they take their medication.

Table 7: Assistive goals.

assistance should be delivered and what technological support would be needed. Table 7 lists on the left column a few categories of activitities, extracted from the taxonomy, and illustrates them; the right column briefly outlines an assistive goal for each example of activity. Beyond the user and their caregiver, analyzing an assistive goal may also involve experts in human-related sciences to refine the user characteristics and needs by administering standardized assessments to determine the user's skill set and deficiency. This knowledge is then used to determine such dimensions as whether the assistance should be context-aware and the type of assistive support that is required (*e.g.,* reminder, task prompting). These steps are inspired by the human-centered approach to developing assistive computing support proposed by Consel [7].

Assistive services. Assistive goals are carried out in practice by developing assistive services that will be deployed in the smart home of the target user. In Table 8, we describe a representative sample of assistive services that we developed; it lists an abbreviated name of the service used later for conciseness, its target activity, and a short description.

As we developed services for our participants, we realized that they consisted of recurring features. This situation is reflected by the analysis of the service descriptions, provided in Table 8. For example, examining meal preparation and vacuuming reveals that both activities are located at home; they must occur during a given time period and at recurring dates; they can be supervised via sensors; reminders can be sent to the user in case the activity is not performed. Let us system-

[7] Charles Consel [2018]. "Assistive computing: a human-centered approach to developing computing support for cognition." In: *2018IEEE/ACM40th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS). IEEE, pp.23–32.*

atize this analysis to identify the commonalities and variabilities of our assistive services.

Analyzing commonalities and variabilities. Because of the features they shared, we approach our assistive services as a program family and analyzed their commonalities and variabilities. This work first allowed to group services into three main categories: indoor, outdoor, home visit. Indoor consists of activities performed at home (*e.g.*, vacuuming, sweeping, showering, preparing meals) and may be supervised via sensors. Outdoor activities may require preparation time; it should eventually lead the user to leave home at a given time, which can be checked via sensors. Home visit occurs indoor and may also involve activities to prepare it. Variabilities are concerned with the time of an event, the time period, the sensor involved, the kind of user interaction, the communication modality with a caregiver, and the task prompter. Commonalities and variabilities are summarized in Table 9.

A taxonomy of technology-supported activities. We have organized the commonalities and variabilities identified previously into a taxonomy (displayed in Figure 13), which classifies technology-supported activities and elicits their characteristics. As such, it can serve as a guide for caregivers to define an activity-supporting service and address the user's needs. At the root of this hierarchy, an activity needs to have a description. It may be recurring and may be supported by actions. The next level introduces a choice between outdoor, indoor, and home visit-related activities. Outdoor activities require a date and a time at which the user is notified to start preparing, as well as a date and a time at which the user is supposed to have departed from home. User departure can be checked via a detector. Indoor activities consist of a date, a time period, and a sensor, if supervision is needed. Finally, home visit-related activities require a date and a time. The leaves of our taxonomy consist of actual activities that inherits the characteristics of the parent levels.

Our taxonomy of technology-supported activities suggests a staged process to define services that could be tooled. This opportunity is explored in the next section.

5.3 *A Wizard for Caregiver Development of Services*

Our aim is to create a tool that 1) covers the taxonomy for technology-supported activities and 2) provides an accessible user interface such that caregivers and clinicians without programming skills can define services that address their care re-

Name	Activity	Service description
PM	Preparing meals	The service measures key user interactions with the environment (fridge, kitchen cabinet, *etc.*) via sensors to detect whether a meal is being prepared within a set time interval, supplied by the user at configuration time. The user and/or a caregiver is notified, when no meal preparation is detected via a tablet notification or a text/email message. The service can also assist the user in preparing a meal by launching a dedicated prompter.
BT	Bedtime routines	Driven by user-declared routines, the service checks whether they are realized by monitoring user interactions with their environment via sensors. When a mismatch is detected the user and/or the caregiver are alerted as in the previous service.
VA	Vacuuming	User is reminded of vaccuming, according to a user-supplied schedule. The vaccum cleaner is equipped with a sensor to check whether it is running, allowing reminders to be sent appropriately. Also, a dedicated task prompter can be launched to assist the user in performing the task.
IR	Ironing	Same as vacuuming. Sensor-equipped iron for context-aware reminders.
SW	sweeping	Same as vacuuming. A sensor is located at a strategic location to detect whether the activity is being performed (*e.g.,* the door of a cabinet containing cleaning items).
IN	Practicing a musical instrument	Same as vacuuming. The musical instrument is equipped with a sensor. Dedicated task prompter can be launched to assist the user in starting setting up the instrument.
SH	Showering	The service notifies the user and/or their caregiver, when no showering activity is detected (via a motion detector), according to the user-supplied schedule.
TR	Taking out the garbage	The service notifies the user and/or their caregiver when the garbage is not taken out, according to the user supplied-schedule. This activity relies on a dedicated sensor, placed at a strategic location.
WO	Going to work	The service sends a notification to the user before and at departure time, according to a user-declared schedule. User departure is checked via sensors. If the user is late, an alert is sent to them and/or their caregiver.
SP	Swimming pool	Same as previous service.
CP	Healthcare-related visit	The service sends a reminder to the user before the time of the appointment. Additionally, a dedicated prompter can be launched to assist the user in preparing the visit (personal care, household chores, *etc.*).
CG	Caregiver visit	Same as previous service with a dedicated task prompter.
ME	Medication taking	A dedicated sensor checks medication is accessed (*e.g.,* the door of a cabinet) at the user-supplied times. An alert is sent to the user, if the activity is not performed. A dedicated task prompter can be launched to guide the user, if needed.

Table 8: Description of services.

	Indoor									Outdoor		Home visit	
	PM	BT	VA	IR	SW	SH	TR	IN	ME	WO	SP	CP	CG
Description	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Begin date	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	×	×	×
End date	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	×	×	×
Recurrence	✓ *	✓ *	P	P	P	✓ *	P	P	P	✓	✓	P	P
Reminder	×	×	×	×	×	×	×	×	×	P	P	✓	✓
Alert	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	×
Preparation date	×	×	×	×	×	×	×	×	×	✓	✓	×	×
Exit date	×	×	×	×	×	×	×	×	×	✓	✓	×	×
Date of reminding	×	×	×	×	×	×	×	×	×	×	×	✓	✓
SMS	P	P	P	P	P	P	P	P	P	P	P	P	P
Email	P	P	P	P	P	P	P	P	P	P	P	P	P
Guiding (Prompter)	P	P	P	P	P	P	P	P	P	P	P	P	P
Supervised (sensor)	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	×	×	×
Departure detection	×	×	×	×	×	×	×	×	×	✓	✓	×	×

P: Possible (depends of the needs) ✓*: mandatory recurrence for such activities as preparing meals and showering.

Table 9: Family of assistive services.

ceiver's needs. As suggested by our taxonomy of technology-supported activities, defining assistive services should be a staged process, allowing the user to specify the characteristics of the target service in a stepwise manner. To match this requirement, our tool has been designed as a wizard, which makes explicit the decomposition of a service definition, reducing the risk of errors [8].

We first discuss the design of our wizard. Then, we illustrate the use of our wizard by creating an indoor activity-supporting service, showing screenshots of our tablet-based Android implementation.

[8] Dmitry Kovalenko [March 2017]. *16 Tips that Will Improve Any Online Form.*

Designing a Wizard

We examine the service characteristics that need to be supplied by the wizard user. Then, we present the task flow underlying our wizard. Finally, we outline a few elements used to design the user interface of our wizard.

User-supplied service characteristics. Our taxonomy (Figure 13) has already made explicit the activity characteristics that need to be supplied by the user of our wizard to define a service. As can be noticed, three categories of services emerge: indoor, outdoor, home visit. We thus revisit the activity characteristics by defining them for service category. Consequently, indoor activities consist of an activity description, a start date and time, an end date and time, a recurrence (optional), a sensor (optional),

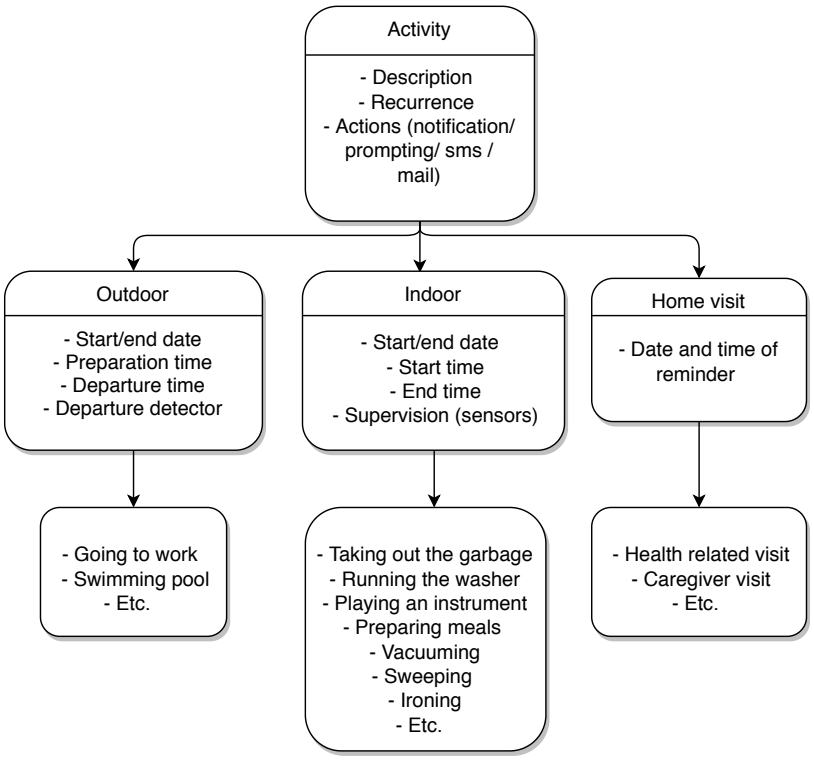


Figure 13: A taxonomy for technology-supported activities.

and an action. A sensor is selected for a service when its activation at a given time period suggests that the activity is being performed (*e.g.,* personal care activity can be assumed when motion is detected in the bathroom in the morning). Different actions can be included in a service; they are detailed in our example below.

Outdoor activities are composed of an activity description, a date and time to start preparing, a date and time to depart from home, a recurrence (optional), and an action to trigger if departure is not detected. Last, home visits consist of a description, the date and time of a reminder, a recurrence (optional), and an action if a preparation activity is not detected.

Task flow. Our taxonomy (Figure 13) suggests a flow of specific information to be supplied and decisions to be taken by the user to define a service. This task flow is shown in Figure 14 with each step represented as a rectangle. The first step is to choose a type of activity: indoor, outdoor, home visit. Then, the user is prompted with category-specific information. Every step requires the user-supplied information to be complete before going to the next step. Note that an indoor activity requires a time interval within which the activity must occur. An outdoor activity also requires a time interval to be defined: the start time is when preparation must begin, whereas the end time is when the user is supposed to depart from home.

In contrast, home visit does not define an interval but a time at which the visit is reminded to the user.

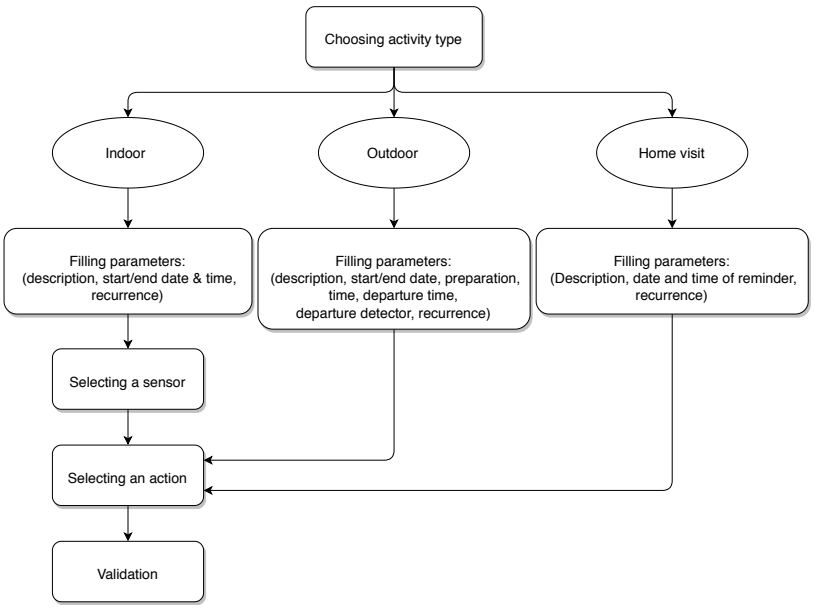


Figure 14: The task flow of our wizard.

User Interface. The general design of the user interface of our wizard follows the usual rules of such a tool: conforming to the users' mental model of the target process, enforcing a clear sequential order of the steps, showing a progress status with numbered steps, allowing navigation buttons to go back and forth in the process, *etc.* We iterated the design of our wizard with caregivers to ensure the activity characteristics were prompted in an order that matched their preference. We also ensured that each wizard step consisted of a self-explanatory title and field names.

Figure 15: Step 1: Choosing the kind of activity to assist.

An Example

Let us explore the details of our wizard by creating an assistive service for doing laundry. The first step, shown in Figure 15, allows the user to choose the indoor activity category. Next, the user is prompted with the parameters of the target activity to be assisted, as shown in Figure 16: the activity description, the date and time, and the frequency. Start/end dates and times are selected via a calendar and a clock, respectively. Frequency is defined via a dedicated menu.

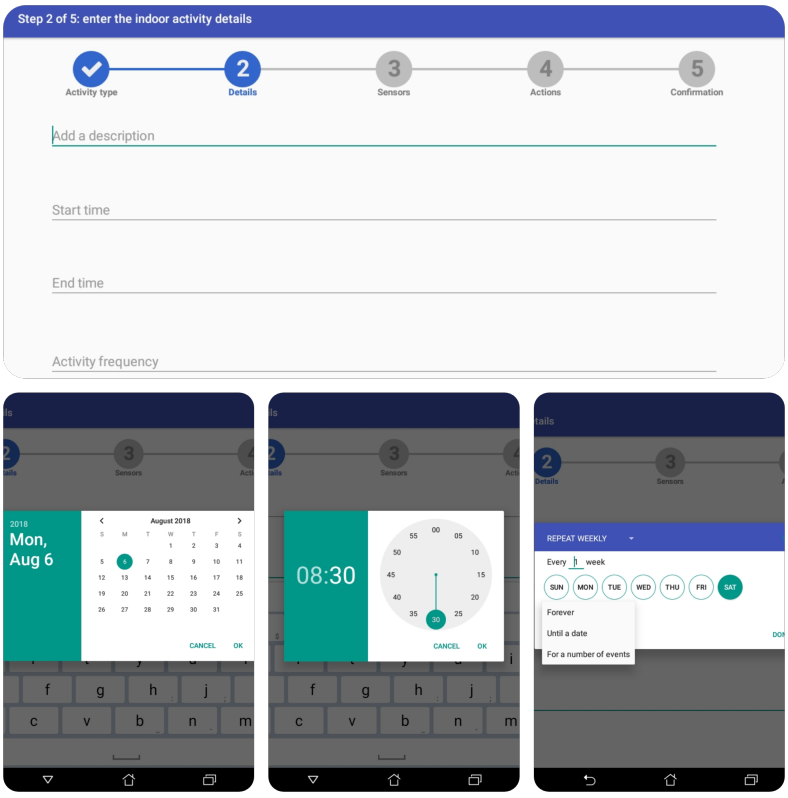


Figure 16: Step 2: Indoor activity details step (description, periodicity, frequency).

The third step, shown in Figure 17, involves deciding whether the activity needs to be monitored. To do so, a sensor category is first selected; there are three categories: 1) sensors attached to electric appliances (iron, coffee machine, washing machine, *etc.*) to detect whether they are running; 2) contact sensors to detect the opening/closing of drawers, cabinet doors, and room/entrance doors; 3) motion sensors to detect a presence in a room or a specific location in a room (depending on the layout). In practice, we have added sensors as new assistive needs were revealed by discussions with participants and six caregivers over various durations of deployment, ranging from one month to a year.

In the fourth step, Figure 18 shows the actions that can be

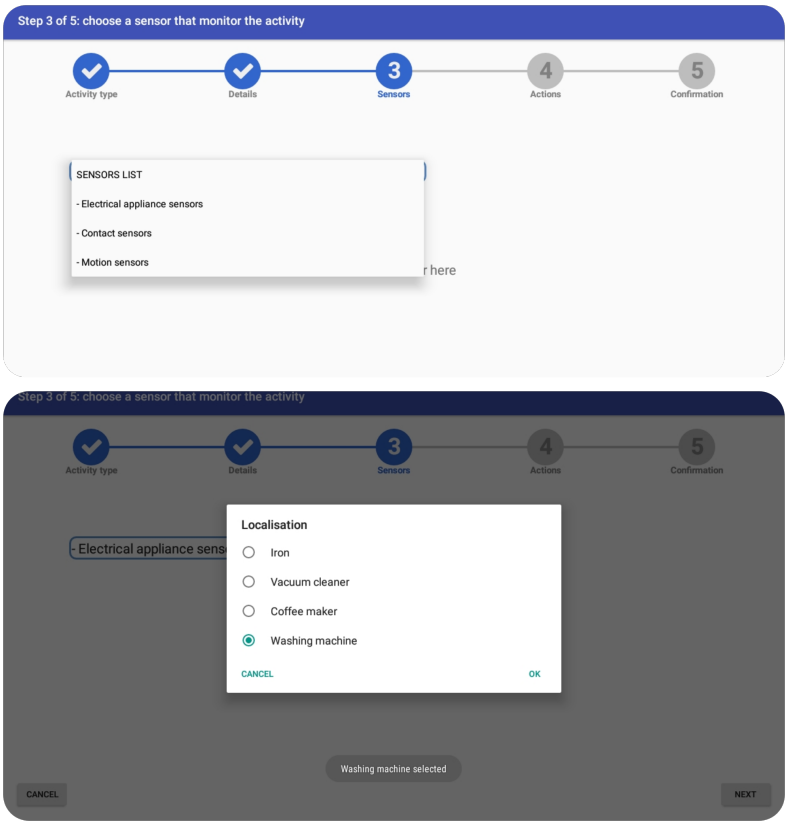


Figure 17: Step 3: choosing a sensor.

associated with an activity. The first action is a reminder to be sent to the user, whose meaning depends on the activity type. For a home visit, it corresponds to a time at which the user needs to get ready. For outdoor activities, one message is issued to inform the user that their preparation should start. An alert is then sent, if the user is still detected at home after the departure time. Indoor activities can trigger an unconditional alert when the time of the activity has arrived, or a conditional alert if the activity is monitored, as is the case in our laundry example. When an alert is issued, it prompts the user for one or more answers, which acknowledge that it has been taken into account.

Note that alerts can be defined as critical or non-critical, depending on the nature of the activity. Non-critical notifications can be ignored by the user, whereas critical ones will repeat the notification until the user responds. A notification is associated to an answer, allowing the service to check whether the user has responded to it. The second and third actions presented in Figure 18 are an email or text message that can be sent to a caregiver in case the activity has not been performed by the user.

Last, Figure 18 shows a menu allowing the wizard user to

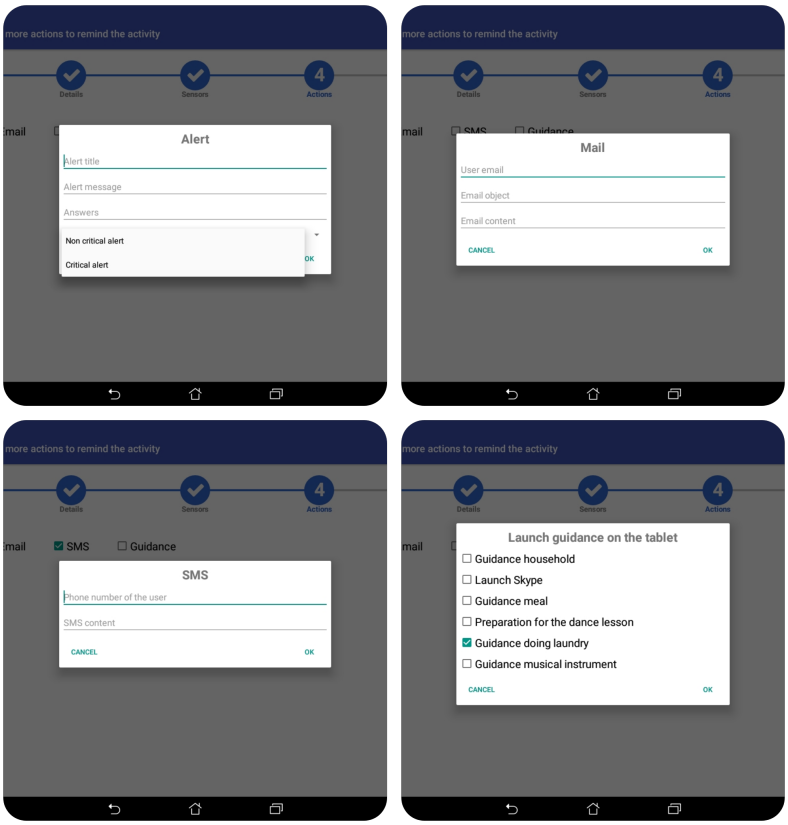


Figure 18: Step 4: choosing one or more actions for the user and their caregiver.

launch a task prompter to guide the user in performing an activity. In practice, this task prompter is launched on a tablet and takes as argument the name of the prompting scenario to invoke.

In the last step of the wizard, shown in Figure 19, the wizard user is presented with a summary of the activity to be assisted.

Implementing our Wizard

Our wizard runs on Android tablets and has been implemented in Java. Tablets allow intuitive touchscreen interface, facilitating the usability of the wizard by caregivers. We used Android's activity transition layout to navigate back and forth between the different wizard forms to be filled by the user. The Android SDK provides a range of UI controls and components to support the implementation of applications such as wizards.

5.4 *Executing Wizard-Defined Services*

We now present the different building blocks that allow wizard-defined services to be executed by a smart home. Our overall system following the wizard stage is displayed in Fig-

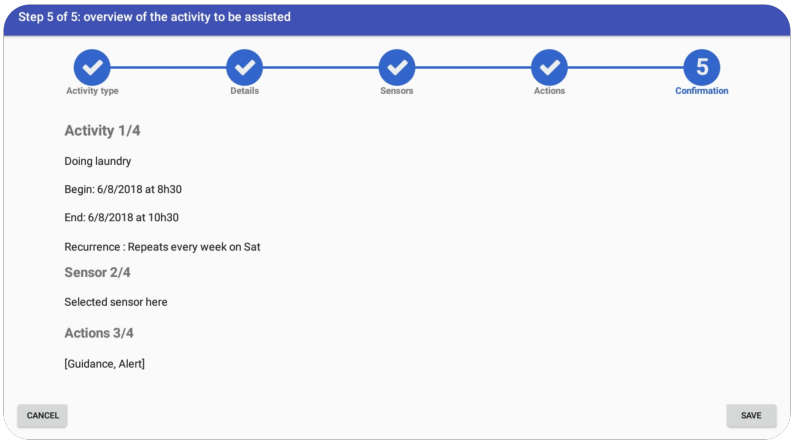


Figure 19: Step 5: validation step.

ure 20 taking the output of our wizard as the starting point. First, we examine what is required to implement our approach in a smart home. Then, we describe how we implemented it using the HomeAssist platform.

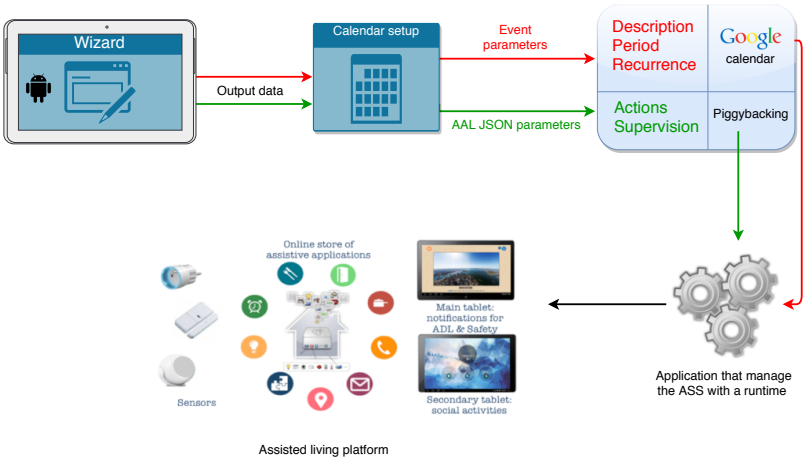


Figure 20: The overall system.

Smart Home Requirements

As suggested by the previous section, executing wizard-defined services revolves around a calendar to manage dates, times and recurrence. In fact, part of the output of our wizard consists of standard calendar event parameter information; this wizard output is denoted by the red arrow in Figure 20. Most calendar provides an API, allowing event to be created with respect to these parameters.

Executing a wizard-defined service still requires to invoke a runtime component to carry out the assistance of the activity, when its calendar event is triggered. The assistance-specific pa-

rameters are denoted by the green output arrow of the wizard (Figure 20) and include information mentioned earlier, such as sensor and notification. These parameters need to be associated with the calendar event and be passed to the runtime component, which is in charge of performing the required actions on the smart home, such as querying a sensor and issuing a notification.

HomeAssist Implementation

Figure 20 presents our implementation based on HomeAssist. As can be noticed, HomeAssist uses Google Calendar, whose API is used by our implementation to manage events. Furthermore, the activity-supporting service characteristics are piggybacked in a calendar event so that the runtime component can extract them when the event is triggered. Specifically, Figure 21 shows the ouput of the wizard (green arrow) in a JSON format for the example of doing laundry, in the context our HomeAssist implementation. Property *MonitoredEvent* is true when the activity is supervised by a sensor, whose name is given by Property *Conditions*. If the activity is not performed by the user, actions to be triggered are listed in Property *Actions*. For example, Property *Android intent* contains the package name of the prompter application, which gets triggered to assist the user in accomplishing the target activity. The last property defines the notification to be issued to the user, including its title and message (Property *non-critical_notification*).

```
{
   "monitoredEvent":true,
   "conditions":{
      "Emeter_Laundry":true
   },
   "actions":{
      "android_intent":{
         "packageName":"com.apps.gk.firstthen"
      },
      "non_critical_notification":{
         "id":"IdNotifier",
         "title":"Turn on the washing machine",
         "text":"It is time to do the laundry.",
         "answer":[
            "Ok"
         ]
      }
   }
}
```

Figure 21: The JSON output of the wizard.

In HomeAssist, smart home applications can be developed and added to a catalog of applications available to users, in the spirit of mobile app platforms. Applications of this catalog

support three main areas: ADLs (*e.g.*, monitoring meal preparation and self-care), user and home safety (*e.g.*, a light path to the bathroom at night and monitoring the stove), and social participation (*e.g.*, simplified email tool and games).

We leveraged this capability by developing an application, dedicated to carry out the assistance of all wizard-defined services. This application subscribes to wizard-related calendar events, extracts piggybacked information from an event field, and provides the assistive support accordingly. For example, in the case of doing laundry, the application checks whether the electric meter of the washing machine is on within the time interval set for this activity. If not, it issues a non-critical notification to the user and launches the prompter on a dedicated tablet.

5.5 Evaluation

Activity coverage and execution equivalence

To evaluate our approach, we used our wizard to define existing activity-supporting services that had been developed manually for older adults, adults with autism, and adults with intellectual disability. In doing so, we wanted to determine whether the wizard could be used to reproduce the development of existing services. This work was quite useful to refine the functionalities of the wizard and ensure that it offered the features needed to cover the existing applications, whose usefulness had already been validated by users and caregivers. Although this first phase allowed us to validate the coverage of the wizard in practice, it did not address the execution of the wizard-defined services. In particular, we still had to show that the execution of wizard-defined services was equivalent to their manually-programmed counterparts. To do so, we developed our special-purpose application in HomeAssist, which is dedicated to execute the wizard-defined services (as explained in Section 5.4). After testing it, we deployed it in the home of our participants to enable wizard-defined services to be executed in real environments. These updated platforms allowed us to validate that the behavior of wizard-defined services was equivalent to their manually-programmed counterparts.

Usability study

Usability tests were conducted with occupational therapists in an apartment laboratory. The goal was to document the perspectives of occupational therapists on the wizard. Indeed,

these clinicians are trained professionals 1) to assess the needs of people living with cognitive impairments and 2) to determine the types of interventions, which can ensure their safety and increase their independence with respect to specific activities [9]. They are also able to anticipate facilitators and obstacles to the facilitators and obstacles to the implementation of new technologies, such as assistive technologies for cognition [10].

Methods. Our usability testing approach was based on two methods: *Cognitive Walkthrough with Users*, and administration of standardized usability questionnaires. The Cognitive Walkthrough with Users is a method that consists of evaluating the usability of an interactive system by constructing different usage scenarios [11]. While they interact with a system, the users are asked to *think aloud*, allowing the experimenter to record their thoughts, feelings and opinions on different aspects of the system being studied. Users perform the tasks of interest after a brief presentation of the experiment. The user's evaluation of the design features of a system is a key factor that determines technology acceptance and is of great importance to software designers [12]. Each occupational therapist was met during a 60-minute session. First, the participants spent 5 minutes introducing themselves to the technology by reading a document presenting the various features of the wizard. Then, they received two clinical vignettes with 2 fictitious patients for whom they had to find solutions using the Wizard application. During this period, subjective data was collected through recording of the participant's voice; objective data (*i.e.*, time spent for each task, number of errors) was collected through recording of the tablet's screen.

Two usability questionnaires (System Usability Scale (SUS) and Attrackdiff) were also administered after the completion of the tasks by each participant. The SUS is a 10-item questionnaire with five response options for respondents; from Strongly agree to Strongly disagree. It is commonly used to assess a wide range of technologies from hardware to mobile applications [13]. The AttrakDiff is a 28-item questionnaire, which assesses user experience through 3 dimensions: Pragmatic Quality (PQ), Hedonic Quality (HQ) and ATTractiveness (ATT) [14]. For both of these questionnaires, psychometric properties such as reliability and validity have been demonstrated.

Results. A total of 5 occupational therapists from different clinical settings in psychogeriatrics agreed to participate in the study, as shown in Table 10. As reported in the literature, 5 participants can lead to the identification of approximately 80% of the usability problems [15].

[9] Walter Wittich et al. [2015]. "Screening for sensory impairment in older adults: Training and practice of occupational therapists in Quebec: Formation et pratique des ergothérapeutes du Québec dans le dépistage des troubles sensoriels chez les personnes âgées." In: *Canadian Journal of Occupational Therapy* 82.5, pp. 283–293.

[10] Susanne Smith Roley et al. [2008]. "Occupational therapy practice framework: domain & process 2nd edition." In: *The American journal of occupational therapy* 62.6, p. 625.

[11] Thomas Mahatody et al. [2010]. "State of the art on the cognitive walkthrough method, its variants and evolutions." In: *Intl. Journal of Human–Computer Interaction* 26.8, pp. 741–785

[12] Tamar Ben-Bassat et al. [2006]. "Economic and subjective measures of the perceived value of aesthetics and usability." In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 13.2, pp. 210–234.

[13] Aaron Bangor et al. [2008]. "An empirical evaluation of the system usability scale." In: *Intl. Journal of Human–Computer Interaction* 24.6, pp. 574–594.

[14] Carine Lallemand et al. [2015]. "Création et validation d'une version française du questionnaire AttrakDiff pour l'évaluation de l'expérience utilisateur des systèmes interactifs." In: *Revue Européenne de Psychologie Appliquée/European Review of Applied Psychology* 65.5, pp. 239–252.

[15] Carl W Turner et al. [2006]. "Determining usability test sample size." In: *International encyclopedia of ergonomics and human factors* 3.2, pp. 3084–3088

Participants	Sex	Age	Degree	Years of experience	Familiar type of software
P1	Female	29	Master	3	Android
P2	Female	44	PhD	17	iOS
P3	Female	49	Bachelor	20	iOS
P4	Female	25	Master	1	iOS
P4	Female	27	Mater	3	iOS

Table 10: Participant's characteristics.

Qualitative data: Subjective data about participants' thoughts, feelings and opinions reveal good usability potential of our wizard, with some suggestions for improvement. One suggestion is related to the way the wizard is supplied information to schedule an activity. Specifically, all of our participants found it difficult to program a recurring activity (*e.g.,* from Monday to Friday). One participant suggested: *"It would be more intuitive to specify that an activity is recurring while setting its date and time than to do it in two phases ."*

Another participant further suggested: *"It would be nice to have access to a small calendar that gives access to all configured reminders because it is easily forgotten"*.

In relation to the way users can configure alerts, participants suggested that an option to make them repeat should be offered since some care receivers may need to receive an alert more than once to ensure appropriate actions are taken. Participants also were uncertain about how to fill the parameters of the alert menu (Figure 18). One participant said:

"It's not obvious to understand what message to put in the alert menu... In particular, It's not clear how to fill 'Answers'; it should show answers by default or be more explicit...". In fact, this field was introduced at the end of our design process and its comprehension was not properly tested with users prior to our study. Since then, it has been changed to take these comments into account.

Another issue noted by the 5 participants was about the sensor options, offered to monitor an activity (Step 3 of the wizard – See Figure 17). Most participants found it too restrictive to use only one sensor for monitoring indoor activities. Indeed, the unique sensor may be activated, and yet, the activity may not be properly completed. For example, one participant suggested that a care receiver may open and close the washing machine door, without loading the laundry, putting the soap, or launching the washing machine. Yet, since the contact sensor of the door was activated, the activity could wronly be considered as completed. A participant suggested:

"It would be interesting to have the possibility to add several sensors to detect an activity. For example, a contact sensor for monitor-

ing the washing machine door and an electric sensor for finding out whether it runs"

Participants also gave suggestions to improve the process of defining services. For example, one participant suggested to use speech recognition to improve efficiency, as most clinicians have time constraints. Another participant suggested to add a fourth type of activity, named "preparation/organizer"; a programmed alert and/or launch of the task prompter would assist the care receivers to organize and prepare their upcoming activities.

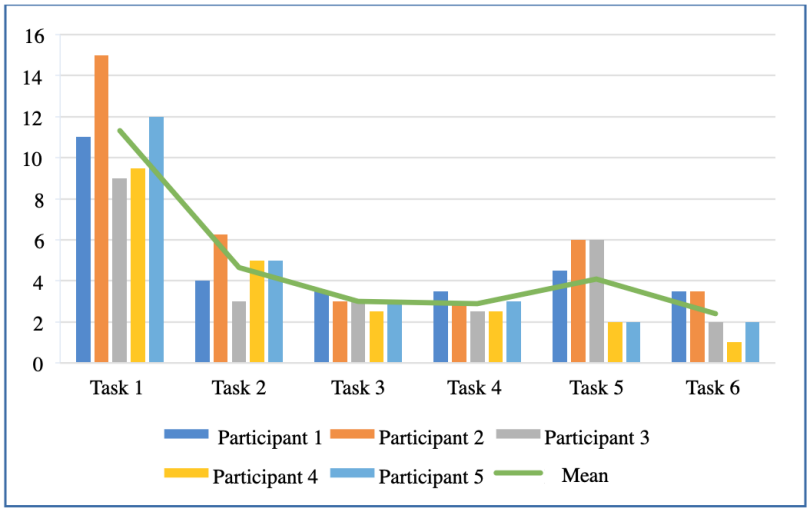


Figure 22: Time (mins) to achieve tasks 1 to 6 for each participant (n = 5).

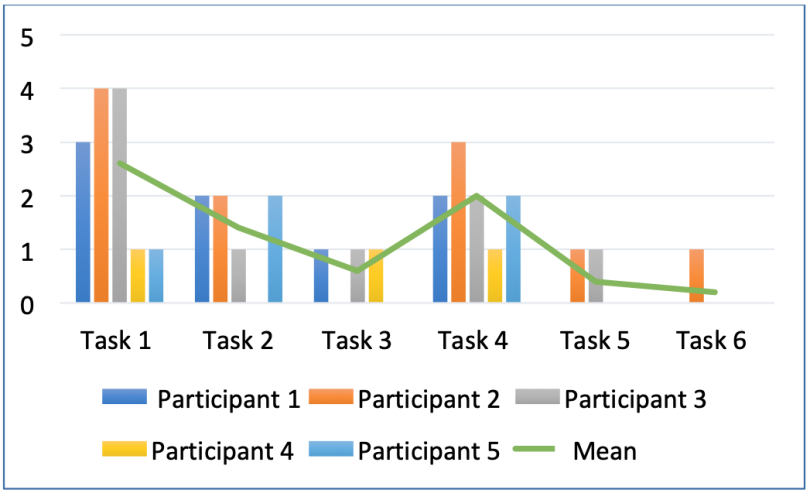


Figure 23: Amount of errors per participant for tasks 1 to 6.

Quantitative data: They were collected while participants completed their tasks using our wizard; their analysis reveals promising findings. Specifically, as shown in Figure 22, the average time to complete each task decreased as participants became more familiar with the application. The number of er-

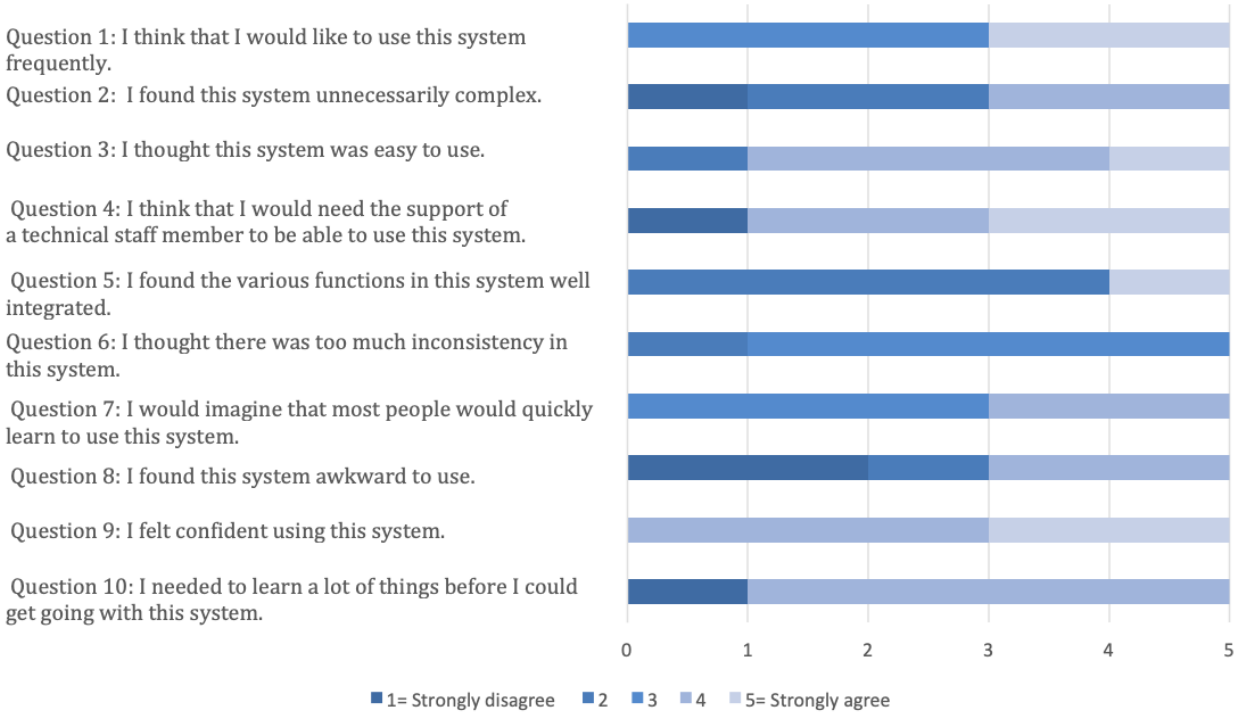


Figure 24: Participant Responses (n = 5) to the System Usability Scale Questionnaire (SUS).

rors per task also decreased, as shown in Figure 23. This trend does not apply to the fourth task, for which all the participants experienced difficulties to select the days of the week for which an activity needed to be scheduled, as discussed earlier. These results suggest that with time and practice, the application becomes easier to use.

Finally, the data collected from the usability questionnaires suggest good usability properties. In particular, Figure 24 shows that the average score of the participants' answers to the Attrackdiff items are positives, except for 3 of them: creativity, practical aspect and human aspect. For the SUS, participants' responses were generally similar for each question, as shown in Figure 25. The only negatives scores matched the issues discussed during the recording of the participants, as mentioned above.

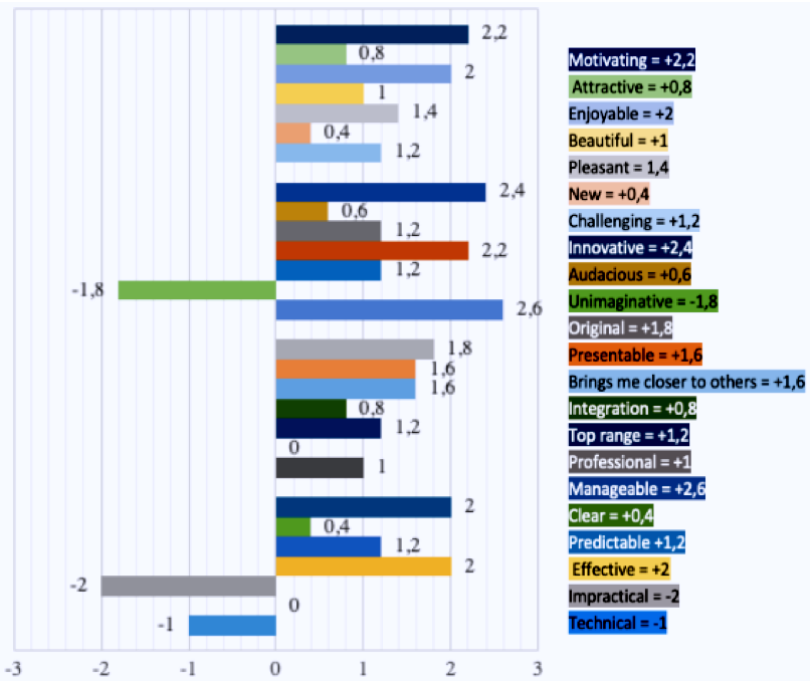


Figure 25: Participant Outcomes (n = 5) to the AT-TRACKDIFF Questionnaire.

Chapter 4: Summary

We have characterized an area of home activities that is needed for independent living and can be supported by smart homes. To address this area, we have introduced a wizard-based approach towards empowering caregivers to develop activity-supporting services, leveraging smart home functionalities. As such, our approach allows the expertise of caregivers to be directly applied to defining assistive support for an individual they care for. We showed how the information gathered by the wizard can be interfaced with a smart home, to carry out the activity-supporting service. We evaluated wizard-defined services by comparing them to manually-programmed services and ensuring that they both had the same behavior. In particular, this evaluation was done by deploying wizard-defined services in the home of participants. We also conducted a usability study of our wizard with professionals. The study showed a good usability potential and an ease of use.

6
Conclusion and Future Work

This chapter details the conclusions of this work. We begin with a discussion on the different contributions we have presented throughout this dissertation. Then, we show how our methods address the challenges identified in the introduction of this thesis in Chapter 1. Finally, we discuss the limitations of each methodology presented and the avenues for future work.

Contents

Overview

- Concluding remarks on the proposed approaches to developing assisted living services.
- Current limitations of our approaches and future work.

This thesis presented three methods of developing assistive services that support independent living. First, we proposed an approach to develop activity recognizers that are accurate in order to improve their effectiveness and acceptability to users and caregivers. This first approach is based on tools that ensure the agile development of service recognizers. Second, we proposed an approach dedicated to monitoring activities on sizeable sensor data. Compared to the first method, this approach raises the level of abstraction for expressing activity detectors, using a domain-specific language (Allen). This DSL provides high-level and concise rules for an even more agile development and also to support reproducible research. Furthermore, this approach provides a visualization tool for experts in aging, displaying a synoptic view of activities on the long term, augmented with information about sensor failures. Although Allen is easier to access than a general or scripting programming language, it remains a language difficult to access for experts in aging who are not programmers. Thereby, to leverage caregivers' expertise, we presented an end-user approach that provides them a step-by-step process to develop assistive services which supports aspects of a daily activity, specific to an older adult. To do so, we introduced a wizard-based interface in order to facilitate the definition of services by end users. In addition, we showed how the caregiver-defined services are uploaded in a real assisted living platform.

6.1 *Discussion*

We now review our approaches in more detail with respect to the key challenges identified in Chapter 1.

To address the challenges for *accurate activity recognizers* introduced in Section 1.1, we provided an approach that: 1) covers the variations of a target activity by abstracting over descriptions reported by users; 2) ensures proper customization with respect to user specificities using a visualization tool. To assess the accuracy of our approach, we showed its inter-individual consistency since we applied it on 6 activity recgonizers and five different user/home configurations. The outputs of our activity recognizers were compared to the activities self-reported by our participants. The results show that 80% of the outputs were confirmed by the user reports. The accuracy of our activity detectors goes up to 88% when considering the more routinized participants.

We addressed the *long-term monitoring* challenges presented

in Section 1.2 by (1) proposing an iterative process that supports a gradual refinement of the analysis of sizeable sensor data. This process allows to reliably detect sensor failures or daily activities; (2) we introduced a long-term visualization tool that provides caregivers a synoptic view of user activities . To assess the accuracy of our approach, we applied our monitoring rules on 5 longitudinal datasets, collected over a period of 12-months. Then, using SDT, the rules outputs were compared against the manual labels of a human expert in activity analysis, which were used as a baseline. As a result, our monitoring rules mostly produced the same interpretations as the human expert.

To address the challenge of *enabling end-user development of service* presented in Section 1.3, we proposed a wizard-based approach that allows caregivers to easily define assistive support for the older aduts they care for. Our finding suggests an ease of use of the wizard by occupational therapists. They have been able to successfully define activity-supporting applications, without programming background.

6.2 *Limitations and Future Work*

Our tooled-methods are a first step towards supporting the development of assisted living applications, and present a number of limitations.

Evaluation. For the proposed methods of this dissertation, we are planning to conduct our experiments with a larger group of participants in their smart environment to assess whether our results scale up. Furthermore, we plan to consider more daily activities to investigate the range of applicability of our methodologies. Another direction for future work is to apply our approaches in other smart home environments. This is important to assess the applicability of our methodologies to a range of smart homes. By using Signal Detection Theory in Chapter 4, we have shown the accuracy of our rule for detecting activities and sensor failures. However, the accuracy of the rules has 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 our approach.

Screening. Another line of work we are exploring aims to link a user's activity data with 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 longitudinal case study (see Chapter 4), which was limited in duration and did not consider the clinical data of its 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.

Activity coverage of the Wizard. In the future, we plan to extend the kind of sensors that can be used in the wizard to detect richer activity contexts than those defined by a unique sensor. In particular, we would like to introduce a high-level notion of sensors that would allow caregivers to exploit activities involving a set of sensor activations, activation durations, etc. In fact, we already introduced this kind of sensors with our departure detector. In practice, not only does this detector monitors the entrance door, but it also checks that there is no motion at home for a while before declaring that the user has departed. For another example, consider a routine for going to bed that may involve motion in the bathroom, followed by motion in the bedroom (see Chapter 3). Needs for such high-level sensors are naturally and promptly expressed by caregivers, as reported by our study, as they discover the potentials of technology to support independent living of individuals they care for. These high-level sensors should be made available in the wizard, as well as simple ones, increasing the coverage of activity-supporting services.

As illustrated in Figure 26, the contributions of our work can be used as a part of a complete activity monitoring system that operates in real time.

For an agile development of services, professional developers can benefit from our iterative development process, illustrated in Chapters 3 and 4. This iterative process supports a stepwise refinement of the analysis of the sensor data, driven by dedicated tools.

For caregivers, we imagine a tool for service development with a high level of abstraction, which can be an improved version of our wizard (with more activity coverage). The wizard can be accompanied by the *interactive configuration* tool that we proposed in Chapter 3, Section 3.4. This can support caregivers to check and adjust the values of activity recognizers parameters with respect to the real sensor data.

For better service reliability, a *maintenance tool* can also be part of the system to detect sensor and platform failures in real time. This will allow technicians to intervene quickly to remedy the failure.

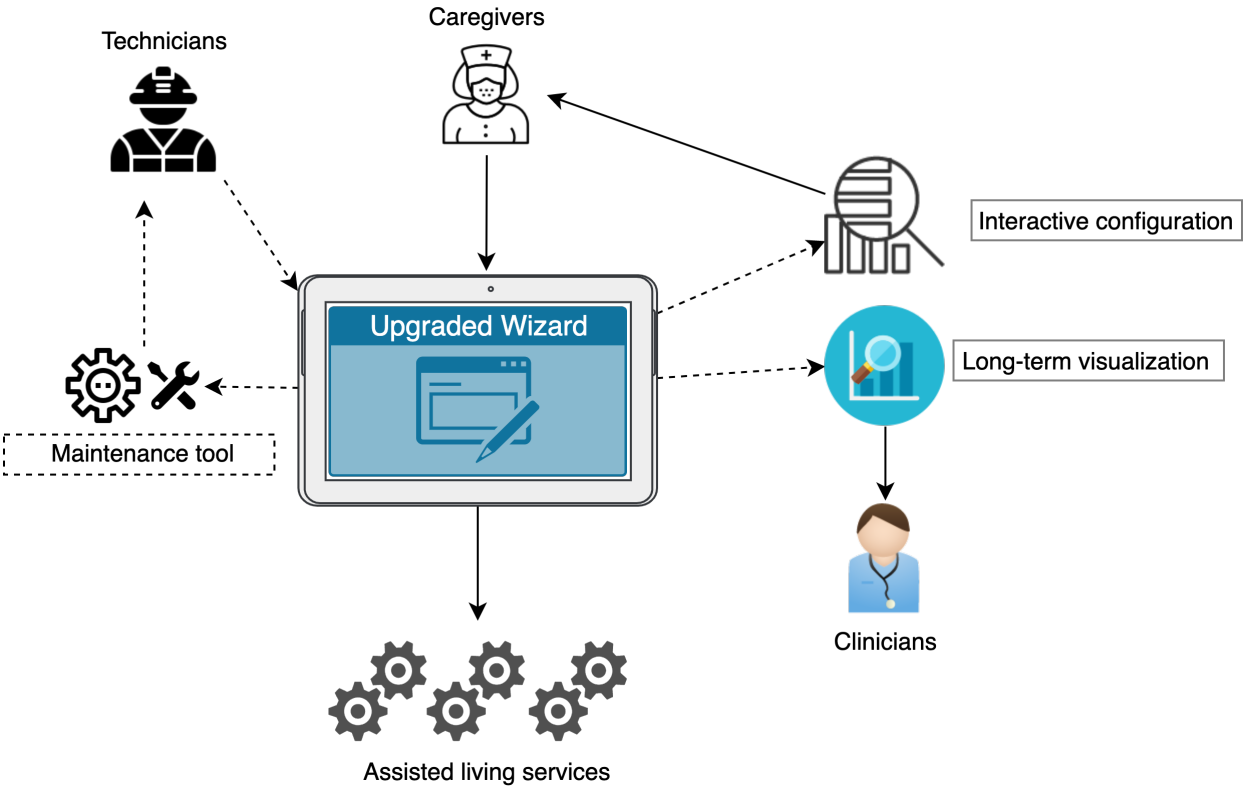


Figure 26: A complete activity monitoring system.

Finally, the *long-term visualization tool* presented in Chapter 4, Section 4.2 can also be integrated into the system to allow clinicians to detect routine deviation of older adults and evaluate the effectiveness of the assisted living platform.

Appendices

Allen Syntax

```
Prog -> Use* Lib Rules?
Use -> "use" id ("[" int "]")? ("(" int ")")?
Lib -> (Def | Let)*
Def -> "def" id ("[" id+("," ) "]")? ("(" id*("," ) ")")? str*
       "=" Context
Lets -> Let*
Let -> "let" id "=" Expr "in"
Rules -> id ":" Context (";" Rules)?
Context -> Lets Expr
Expr -> Prod "|" Expr | Prod
Prod -> Comp "&" Prod | Comp
Comp -> Expr1 (">=!"|"<="|">="|">!"|">!!"|"<"|">") Int | Expr1
Expr1 -> true | false | "~" Expr1 | "(" Expr ")" | str
      | id ("[" Int+("," ) "]")? ("(" Expr*("," ) ")")?
Int -> Int1 ("+"|"-") Int | Int1
Int1 -> id | ts | int ("hr" | "min" | "sec")?
```

Figure 27: Syntax of the Allen language.

Static Table of Interactions

```
{
    "presence": {"kind": "Presence",
        "values": ["true", "false"]},
    "door": {"location": "Entrance","kind": "Door",
        "values": ["open", "close"]},
  "cupboard": {"location": "Kitchen", "kind": "Cupboard",
    "values": ["open", "close"]},
  "fridge": {"location": "Kitchen", "kind": "Fridge",
    "values": ["open", "close"]},
  "stove": {"location": "Kitchen", "kind": "Stove",
    "values": ["on", "off"]},
  "juicer": {"location": "Kitchen", "kind": "Juicer",
    "values": ["on", "off"]},
  "toaster": {"location": "Kitchen", "kind": "Toaster",
    "values": ["on", "off"]},
  "meatcleaver": {"location": "Kitchen", "kind": "Meatcleaver",
    "values": ["on", "off"]},
  "microwave": {"location": "Kitchen", "kind": "Microwave",
    "values": ["on", "off"]},
  "kettle": {"location": "Kitchen", "kind": "Kettle",
    "values": ["on", "off"]},
  "coffeeMaker": {"location": "Kitchen", "kind": "CoffeeMaker",
    "values": ["on", "off"]},
  "nightTime": {"location": "Night", "kind": "Calendar",
    "values": ["begin", "end"]},
  "dinnerTime": {"location": "Dinner", "kind": "Calendar",
    "values": ["begin", "end"]},
  "lunchTime": {"location": "Lunch", "kind": "Calendar",
    "values": ["begin", "end"]},
  "breakfastTime": {"location": "Breakfast", "kind": "Calendar",
    "values": ["begin", "end"]},
  "bedTime": {"location": "Bed", "kind": "Calendar",
    "values": ["begin", "end"]},
  "dressingTime": {"location": "Dressing", "kind": "Calendar",
    "values": ["begin", "end"]},
  "wakeUpTime": {"location": "WakeUp", "kind": "Calendar",
    "values": ["begin", "end"]}
}
```

Figure 28: Complete static table of interactions in HomeAssist.

Bibliography

Frances K Aldrich (2003). "Smart homes: past, present and future." In: *Inside the smart home*. Springer, pp. 17–39.

Aaron Bangor, Philip T Kortum, and James T Miller (2008). "An empirical evaluation of the system usability scale." In: *Intl. Journal of Human–Computer Interaction* 24.6, pp. 574–594. DOI: 10.1080/10447310802205776.

Rafik Belloum (2020). "End-user Development of Activity-Supporting Services for Smart Homes." In: *In Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom Workshops). Austin, Texas. 2020*.

Rafik Belloum, Charles Consel, and Nic Volanschi (2020). "A Tool-Based Methodology For Long-Term Activity Monitoring." In: *PETRA'20-Pervasive Technologies Related to Assistive Environments*.

Tamar Ben-Bassat, Joachim Meyer, and Noam Tractinsky (2006). "Economic and subjective measures of the perceived value of aesthetics and usability." In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 13.2, pp. 210–234. DOI: 10.1145/1165734.1165737.

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." In: *The International Journal of Aging and Human Development* 77.4, pp. 309–329. DOI: 10.2190/AG.77.4.c.

Benjamin Bertran, Julien Bruneau, Damien Cassou, Nicolas Loriant, Emilie Balland, and Charles Consel (January 2014). "DiaSuite: A tool suite to develop Sense/Compute/Control applications." en. In: *Science of Computer Programming* 79, pp. 39–51. ISSN: 01676423. DOI: 10.1016/j.scico.2012.04.001. (Visited on 04/22/2020).

Julia Brich, Marcel Walch, Michael Rietzler, Michael Weber, and Florian Schaub (2017). "Exploring end user programming needs in home automation." In: *ACM Transactions on*

Computer-Human Interaction (TOCHI) 24.2, p. 11. DOI: 10.1145/3057858.

A.J. Bernheim Brush, Bongshin Lee, Ratul Mahajan, Sharad Agarwal, Stefan Saroiu, and Colin Dixon (2011). "Home automation in the wild: challenges and opportunities." en. In: *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*. Vancouver, BC, Canada: ACM Press, p. 2115. ISBN: 9781450302289. DOI: 10.1145/1978942.1979249. (Visited on 04/22/2020).

Stefan Parry Carmien and Gerhard Fischer (2008). "Design, adoption, and assessment of a socio-technical environment supporting independence for persons with cognitive disabilities." en. In: *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*. Florence, Italy: ACM Press, p. 597. ISBN: 9781605580111. DOI: 10.1145/1357054.1357151. (Visited on 04/22/2020).

Loïc Caroux, Charles Consel, Lucile Dupuy, and Hélène Sauzeon (2018). "Towards context-aware assistive applications for aging in place via real-life-proof activity detection." In: *Journal of ambient intelligence and smart environments* 10.6, pp. 445–459. DOI: 10.3233/AIS-180505.

Loïc Caroux, Charles Consel, Lucile Dupuy, and Hélène Sauzéon (October 2014). "Verification of Daily Activities of Older Adults: A Simple, Non-Intrusive, Low-Cost Approach." In: *ASSETS - The 16th International ACM SIGACCESS Conference on Computers and Accessibility*. Rochester, NY, United States, pp. 43–50. DOI: 10.1145/2661334.2661360.

Liming Chen, Chris D. Nugent, and Hui Wang (June 2012). "A Knowledge-Driven Approach to Activity Recognition in Smart Homes." In: *IEEE Transactions on Knowledge and Data Engineering* 24.6, pp. 961–974. ISSN: 1041-4347. DOI: 10.1109/TKDE.2011.51. (Visited on 04/22/2020).

Charles Consel (2018). "Assistive computing: a human-centered approach to developing computing support for cognition." In: *2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, pp. 23–32. DOI: 10.1145/3183428.3183431.

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, pp. 129–140.

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 be-*

havior datasets to advance HCI and ubiquitous computing research, pp. 1–7.

Joelle Coutaz and James L. Crowley (April 2016). "A First-Person Experience with End-User Development for Smart Homes." In: *IEEE Pervasive Computing* 15.2, pp. 26–39. ISSN: 1536-1268. DOI: 10.1109/MPRV.2016.24. (Visited on 04/22/2020).

Sara J Czaja, Ruth A Weber, and Sankaran N Nair (1993). "A human factors analysis of ADL activities: A capability-demand approach." In: *Journal of Gerontology* 48.Special_Issue, pp. 44–48.

Krzysztof Czarnecki, Ulrich W Eisenecker, and Krysztof Czarnecki (2000). *Generative programming: methods, tools, and applications*. Vol. 16. Addison Wesley Reading.

Barnan Das, Diane J. Cook, Maureen Schmitter-Edgecombe, and Adriana M. Seelye (October 2012). "PUCK: an automated prompting system for smart environments: toward achieving automated prompting—challenges involved." en. In: *Personal and Ubiquitous Computing* 16.7, pp. 859–873. ISSN: 1617-4909, 1617-4917. DOI: 10.1007/s00779-011-0445-6. (Visited on 04/22/2020).

Prafulla N. Dawadi, Diane J. Cook, and Maureen Schmitter-Edgecombe (November 2013). "Automated Cognitive Health Assessment Using Smart Home Monitoring of Complex Tasks." In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 43.6, pp. 1302–1313. ISSN: 2168-2216, 2168-2232. DOI: 10.1109/TSMC.2013.2252338. (Visited on 04/22/2020).

Prafulla N. Dawadi, Diane J. Cook, Maureen Schmitter-Edgecombe, and Carolyn Parsey (August 2013). "Automated assessment of cognitive health using smart home technologies." In: *Technology and Health Care* 21.4, pp. 323–343. ISSN: 09287329, 18787401. DOI: 10.3233/THC-130734. (Visited on 04/22/2020).

Richard Bernard Dever (1988). *Community living skills: A taxonomy.* American association on mental retardation.

Anind K Dey, Timothy Sohn, Sara Streng, and Justin Kodama (2006). "iCAP: Interactive prototyping of context-aware applications." In: *International Conference on Pervasive Computing*. Springer, pp. 254–271.

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." In: *Frontiers in aging neuroscience* 9, p. 302. DOI: 10.3389/fnagi.2017.00302.

Nancy ElHady and Julien Provost (June 2018). "A Systematic Survey on Sensor Failure Detection and Fault-Tolerance in Ambient Assisted Living." en. In: *Sensors* 18.7, p. 1991. ISSN: 1424-8220. DOI: 10.3390/s18071991. (Visited on 04/22/2020).

Arthur D. Fisk, Sara J. Czaja, Wendy A. Rogers, Neil Charness, Sara J. Czaja, and Joseph Sharit (November 2018). *Designing for Older Adults: Principles and Creative Human Factors Approaches, Second Edition*. en. oth ed. CRC Press. ISBN: 9780429195914. DOI: 10.1201/9781420080681. (Visited on 04/22/2020).

David A Gold (2012). "An examination of instrumental activities of daily living assessment in older adults and mild cognitive impairment." In: *Journal of clinical and experimental neuropsychology* 34.1, pp. 11–34. DOI: 10.1080/13803395.2011.614598.

Trisha Greenhalgh, Joe Wherton, Paul Sugarhood, Sue Hinder, Rob Procter, and Rob Stones (2013). "What matters to older people with assisted living needs? A phenomenological analysis of the use and non-use of telehealth and telecare." In: *Social science & medicine* 93, pp. 86–94. DOI: 10.1016/j.socscimed.2013.05.036.

Xin Hong and Chris D. Nugent (March 2013). "Segmenting sensor data for activity monitoring in smart environments." en. In: *Personal and Ubiquitous Computing* 17.3, pp. 545–559. ISSN: 1617-4909, 1617-4917. DOI: 10.1007/s00779-012-0507-4. (Visited on 04/22/2020).

Mohammad Anwar Hossain (2014). "Perspectives of human factors in designing elderly monitoring system." In: *Computers in Human Behavior* 33, pp. 63–68. ISSN: 0747-5632. DOI: https://doi.org/10.1016/j.chb.2013.12.010.

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, pp. 215–225. DOI: 10.1145/2750858.2805830.

Jeffrey A Kaye, Shoshana A Maxwell, Nora Mattek, Tamara L Hayes, Hiroko Dodge, Misha Pavel, Holly B Jimison, Katherine Wild, Linda Boise, and Tracy A Zitzelberger (2011). "Intelligent systems for assessing aging changes: home-based, unobtrusive, and continuous assessment of aging." In: *Journals of Gerontology Series B: Psychological Sciences and Social Sciences* 66.suppl_1, pp. i180–i190. DOI: 10.1093/geronb/gbq095.

Dmitry Kovalenko (March 2017). *16 Tips that Will Improve Any Online Form*. URL: https://uxplanet.org/the-18-must-do-principles-in-the-form-design-fe89d0127c92.

Carine Lallemand, Vincent Koenig, Guillaume Gronier, and Romain Martin (2015). "Création et validation d'une version française du questionnaire AttrakDiff pour l'évaluation de l'expérience utilisateur des systèmes interactifs." In: *Revue*

Européenne de Psychologie Appliquée/European Review of Applied Psychology 65.5, pp. 239–252. DOI: 10.1016/j.erap.2015.08.002.

M. P. Lawton and E. M. Brody (September 1969). "Assessment of Older People: Self-Maintaining and Instrumental Activities of Daily Living." en. In: *The Gerontologist* 9.3 Part 1, pp. 179–186. ISSN: 0016-9013, 1758-5341. DOI: 10.1093/geront/9.3_Part_1.179. (Visited on 04/22/2020).

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, pp. 483–500.

Thomas Mahatody, Mouldi Sagar, and Christophe Kolski (2010). "State of the art on the cognitive walkthrough method, its variants and evolutions." In: *Intl. Journal of Human–Computer Interaction* 26.8, pp. 741–785.

Marjan Mernik, Jan Heering, and Anthony M Sloane (2005). "When and how to develop domain-specific languages." In: *ACM computing surveys (CSUR)* 37.4, pp. 316–344. DOI: 10.1145/1118890.1118892.

Daniel G Morrow and Wendy A Rogers (2008). "Environmental support: An integrative framework." In: *Human Factors* 50.4, pp. 589–613.

Johan Ormel, Frühling V Rijsdijk, Mark Sullivan, Eric Van Sonderen, and Gertrudis IJM Kempen (2002). "Temporal and reciprocal relationship between IADL/ADL disability and depressive symptoms in late life." In: *The Journals of Gerontology Series B: Psychological Sciences and Social Sciences* 57.4, P338–P347. DOI: 10.1093/geronb/57.4.P338.

David Lorge Parnas (1976). "On the design and development of program families." In: *IEEE Transactions on software engineering* 1, pp. 1–9. DOI: 10.1109/TSE.1976.233797.

Parisa Rashidi and Alex Mihailidis (2013). "A survey on ambient-assisted living tools for older adults." In: *IEEE journal of biomedical and health informatics* 17.3, pp. 579–590. DOI: 10.1109/JBHI.2012.2234129.

Jennifer Reijnders, Caroline van Heugten, and Martin van Boxtel (2013). "Cognitive interventions in healthy older adults and people with mild cognitive impairment: a systematic review." In: *Ageing research reviews* 12.1, pp. 263–275.

M. R. Reisinger, J. Schrammel, and P. Fröhlich (2017). "Visual languages for smart spaces: End-user programming between data-flow and form-filling." In: *2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pp. 165–169. DOI: 10.1109/VLHCC.2017.8103464.

Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, et al. (2009). "Scratch: programming for all." In: *Communications of the ACM* 52.11, pp. 60–67. DOI: 10.1145/1592761.1592779.

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, pp. 427–431.

Susanne Smith Roley, Cynthia J Barrows, L Susan Brownrigg OTR, Deanna Iris Sava, L Vibeke Talley OTR, BS Kristi Voelkerding, L COTA, Emily Smith MOT, MS Pamela Toto, OTR Sarah King MOT, et al. (2008). "Occupational therapy practice framework: domain & process 2nd edition." In: *The American journal of occupational therapy* 62.6, p. 625. DOI: 10.5014/ajot.62.6.625.

A. M. Jehad Sarkar, Young-Koo Lee, and Sungyoung Lee (2010). "ARHMAM: an activity recognition system based on hidden Markov minded activity model." en. In: *Proceedings of the 4th International Conference on Uniquitous Information Management and Communication - ICUIMC '10*. Suwon, Republic of Korea: ACM Press, p. 1. ISBN: 9781605588933. DOI: 10.1145/2108616.2108702. (Visited on 04/22/2020).

Adriana M. Seelye, Maureen Schmitter-Edgecombe, Diane J. Cook, and Aaron Crandall (April 2013). "Naturalistic Assessment of Everyday Activities and Prompting Technologies in Mild Cognitive Impairment." en. In: *Journal of the International Neuropsychological Society* 19.4, pp. 442–452. ISSN: 1355-6177, 1469-7661. DOI: 10.1017/S135561771200149X. (Visited on 04/22/2020).

Harold Stanislaw and Natasha Todorov (March 1999). "Calculation of signal detection theory measures." en. In: *Behavior Research Methods, Instruments, & Computers* 31.1, pp. 137–149. ISSN: 0743-3808, 1532-5970. DOI: 10.3758/BF03207704. (Visited on 04/22/2020).

Alistair Sutcliffe (2005). "Evaluating the costs and benefits of end-user development." In: *ACM SIGSOFT Software Engineering Notes*. Vol. 30. 4. ACM, pp. 1–4.

Elizabeth A Townsend and Helene J Polatajko (2007). "Advancing an occupational therapy vision for health, well-being, and justice through occupation." In: *Ottawa, ON: CAOT Publications ACE.*

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, pp. 144–151.

Khai N Truong, Elaine M Huang, and Gregory D Abowd (2004). "CAMP: A magnetic poetry interface for end-user programming of capture applications for the home." In: *International Conference on Ubiquitous Computing*. Springer, pp. 143–160.

Carl W Turner, James R Lewis, and Jakob Nielsen (2006). "Determining usability test sample size." In: *International encyclopedia of ergonomics and human factors* 3.2, pp. 3084–3088.

TLM Van Kasteren, Gwenn Englebienne, and Ben JA Kröse (2010). "Activity recognition using semi-markov models on real world smart home datasets." In: *Journal of ambient intelligence and smart environments* 2.3, pp. 311–325. DOI: 10.3233/AIS-2010-0070.

Nic Volanschi, Adrien Carteron, and Charles Consel (October 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)*. Guangzhou, China: IEEE, pp. 480–489. ISBN: 9781538693803. DOI: 10.1109/SmartWorld.2018.00108. (Visited on 04/22/2020).

Nic Volanschi, Bernard Serpette, Adrien Carteron, and Charles Consel (December 2018). "A Language for Online State Processing of Binary Sensors, Applied to Ambient Assisted Living." en. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2.4, pp. 1–26. ISSN: 2474-9567, 2474-9567. DOI: 10.1145/3287070. (Visited on 04/22/2020).

Walter Wittich, Elizabeth A Barstow, Jonathan Jarry, and Aliki Thomas (2015). "Screening for sensory impairment in older adults: Training and practice of occupational therapists in Quebec: Formation et pratique des ergothérapeutes du Québec dans le dépistage des troubles sensoriels chez les personnes âgées." In: *Canadian Journal of Occupational Therapy* 82.5, pp. 283–293. DOI: 10.1177/0008417415573076.

Kristina Yordanova, Stefan Lüdtke, Samuel Whitehouse, Frank Krüger, Adeline Paiement, Majid Mirmehdi, Ian Craddock, and Thomas Kirste (February 2019). "Analysing Cooking Behaviour in Home Settings: Towards Health Monitoring." en. In: *Sensors* 19.3, p. 646. ISSN: 1424-8220. DOI: 10.3390/s19030646. (Visited on 04/22/2020).