The Journal of Systems and Software 104 (2015) 166–178

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

A semantic approach for designing Assistive Software Recommender systems

Elena Gómez-Martínez[a], Marino Linaje[b], Fernando Sánchez-Figueroa[b], Andrés Iglesias-Pérez[c], Juan Carlos Preciado[b], Rafael González-Cabero[d], José Merseguer[a,*]

[a] Departamento de Informática e Ingeniería de Sistemas, Universidad de Zaragoza, Maria de Luna, 50018 Zaragoza, Spain
[b] Escuela Politécnica de Cáceres, Universidad de Extremadura, Spain
[c] Ilunion Accesibilidad Estudios y Proyectos, Spain
[d] Ontology Engineering Group, Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Spain

ARTICLE INFO

Article history:
Received 7 July 2014
Revised 10 February 2015
Accepted 9 March 2015
Available online 11 March 2015

Keywords:
Software design
Assistive Software
Software non-functional evaluation

ABSTRACT

Assistive Software offers a solution for people with disabilities to manage specialized hardware, devices or services. However, these users may have difficulties in selecting and installing Assistive Software in their devices for managing smart environments. This paper addresses the requirements of these kinds of systems and their design in the context of interoperability architectures. Our solution follows a semantic approach, for which ontologies are a key. The paper also presents an implementation of our design proposal, i.e., a real and usable system which is evaluated according to a set of functional and non-functional requirements here proposed.

1. Introduction

Universal access continues to be a critical quality target for Information and Communication Technologies (ICTs), as Stephanidis (2001) stated. This is especially important in industrial societies where there is a growing number of people with special needs,[1] including those with aging-related conditions. Indeed, ICTs may require particular skills and abilities to interact with platforms, wireless communication systems and smart devices such as kiosks or ATMs.

Developing universally accessible smart environments is hard in terms of effort and required knowledge (Zimmermann and Vanderheiden, 2008). As an alternative, Assistive Software (AS from now on) provides an easy and feasible solution. AS represents software products specifically designed for people with some disability that is used to increase their ability to manage information in a digital device. AS therefore makes it easier to use ICT devices. This paper is mainly devoted to smart environments, e.g., the smart home (Margetis et al., 2012). For example, a blind person could use AS installed in her/his smartphone for managing mainstream software to control a smart TV or an air-conditioning system.

AS products can be selected in different ways. For example, simply using trial and error by (1) examining a user interface to determine whether it is accessible or not for a given disability (e.g., blindness), (2) finding an AS product that claims to solve the particular interaction issue, e.g., exploring assistive technology repositories such as EASTIN,[2] (3) installing it, (4) returning to step 2 if the AS does not solve the interaction issue and so on. Using this manual form of AS selection, the user spends time and money testing AS products that in the end may not effectively solve the problem. Another interesting possibility is the use of assessment services (Andrich et al., 2013a; 2013b). However, difficulties may arise in finding an Assistive Technology professional, e.g., in the very moment of browsing for finding the AS product. To address these issues, AS Recommender systems (ASR systems from now on) have been developed to help users in making decisions automatically and timely. An ASR system selects the most suitable AS for a specific context using as inputs the needs and preferences of the user, such as privacy, type of device used or type of disability.

This paper deals with the design of ASR systems and the requirements they should address. The design solution presented here is able to select the most suitable AS automatically, following a semantic approach. Indeed, the paper presents the conceptualization of an

* Corresponding author. Tel.: +34685198175.
E-mail addresses: elena.gomez.martinez@upm.es (E. Gómez-Martínez), mlinaje@unex.es (M. Linaje), fernando@unex.es (F. Sánchez-Figueroa), andresip@unex.es (A. Iglesias-Pérez), jcpreciado@unex.es (J.C. Preciado), rgonza@fi.upm.es (R. González-Cabero), jmerse@unizar.es, jmerse@gmail.com (J. Merseguer).

[1] The terminology used in the paper as regards the disability field conforms to the International Classification of Functioning, Disability and Health (World Health Organization, 2001).

[2] http://www.eastin.eu/

http://dx.doi.org/10.1016/j.jss.2015.03.009
0164-1212/

ontology for AS selection. Following the design guidelines, we implemented a Knowledge Base for the ontology and a real and usable ASR system, which is presented in the paper. The system successfully deals with non functional requirements such as response time and scalability.

To cope with complex environment necessities, ASR systems can be deployed in existing interoperability architectures. Such architectures allow access to heterogeneous data sources through semantically enriched services, e.g. using ontologies. For example, the SAPHIRE (Nee et al., 2008) interoperability architecture accesses disparate data sources to retrieve patient-specific information through Web services using standard medical ontologies. In our case, the architecture ensures that (a) the user can interact with a controller device,[3] (b) the target devices or services can publish their user interfaces, (c) the controller device capabilities can be shown and (d) the disability of the user is managed by the architecture. The last two conditions are mandatory for a fully automatic ASR system, but optional for a semi-automatic selection. The ASR system presented here has been deployed in the context of the (INREDIS, 2011a) (INterfaces for RElations between Environment and people with DISabilities) interoperability architecture where it has also been evaluated.

The paper is organized as follows. Section 2 establishes the requirements for an ASR system. Sections 3 and 4 lay the design foundations for a semantic ASR system. Section 5 presents the ASR system we have developed and explains how it was deployed in INREDIS. Section 6 evaluates the ASR system in real scenarios. Section 7 describes related works. Finally, Section 8 outlines the conclusions and future work.

2. System requirements

Tom has an impairment; specifically he is a blind user. He feels comfortable and safer carrying his smartphone, especially when he is out of his native Spain. After a long trip Tom has just arrived at the hotel in Tokyo where he has a reservation. It is his first time in this hotel. He enters the hotel room and his smartphone vibrates. It shows him some devices to interact with, such as the TV or the air-conditioning system (AC). He feels a little bit hot, so he will need to operate the AC through his smartphone to adjust the temperature and eventually to switch off the timer, for instance an hour later. This is an application scenario for an Assistive Software Recommender (ASR) system. Tom needs to automatically set up interfaces in his smartphone to operate the appliances in the hotel room.

The requirements for ASR systems were carefully studied in the INREDIS (2011a) project. At the beginning of the project more than 1000 end-users were asked through questionnaires and interviews to list their needs and preferences. Specifically, 400 telephone surveys and 597 online surveys were carried out (257 with deaf person people and 340 with people with other disabilities). The respondents were randomly selected among ONCE[4] members. Moreover, seven discussion groups were set up for people with different impairments. In addition, 15 open interviews were carried out with professionals, as "key informants", on different profiles of disability. All this information was useful for developing an initial prototype for self-detecting user's needs and capabilities according to ISO standard 24756:2009 (ISO/IEC, 2009). Our conclusion is that an ASR system should meet the following requirements:

Table 1
System requirements.

Requirement	Description
FR1	Detect "accessibility issues" for users with disabilities
FR2	Support anonymous and profile-based requests
FR3	Provide a weighted list of Assistive Software products automatically
FR4	Advise the user whether the selected AS is compliant with available data protection laws
FR5	Incorporate self-learning capabilities according to users' criteria
FR6	Install the selected Assistive Software automatically
FR7	Adapt to the needs of the user
NFR1	Be flexible, easy to use and communicative according to Nielsen principles (Nielsen, 1993)
NFR2	Have a response time according to Nielsen principles (Nielsen, 1993)
NFR3	Be scalable in well-defined environments (e.g., smart home and facilities)

Functional requirements (FR) define the scope of the system from the user point of view. The user will select the AS of her/his choice from a list (FR3, FR6). This list is built considering the needs and preferences of the user (FR1), who could also make requests to the system anonymously through generic profiles (FR2). FR5 confers learning capabilities on the system, from previous user selections.

Non-functional requirements (NFR) shape the system operation mode. In this respect, the main concern is to create an appropriate environment in which people with disabilities and elderly people can operate. In the aforementioned interviews, ONCE experts advised us that the Nielsen principles (NFR1 and NFR2) were a major source of satisfaction. Regarding system response time (NFR2), Nielsen establishes that:

- 0.1 s is about the limit for having the user feel that the system is reacting instantaneously.
- 1.0 s is about the limit for the user's flow of thought to stay uninterrupted, even though the user will notice the delay.
- 10 s is about the limit for keeping the user's attention focused on the dialogue. Users should be given feedback indicating when the computer expects to be done.

Although the target audience in our system has special needs, the response times must be similar to those for users without these needs not taking into account the time spent by disabled people in operating the target device.[5] Then, for the ASR system the expected response times should be within these intervals. Regarding scalability (NFR3), an ASR system should scale within the architecture where it is integrated, in our case the INREDIS architecture. INREDIS was projected for a wide range of real world scenarios with a high number of users. Examples of scenarios where INREDIS has been deployed are leisure services (location and purchasing tickets for events), smart homes (Sainz et al., 2011), urban networking (Giménez et al., 2012), social networks (Murua et al., 2011) or banking services (ATMs; Pous et al., 2012).

Our functional requirements can be addressed at design level through several processes of knowledge management, which are described in Section 4. These processes rely on an ontology, described in Section 3. Table 2 matches each requirement with the process or processes that address it and the part of the ontology required. An assessment of the functional and non functional requirements is reported in Section 6.

[3] A *controller device* (e.g., a smartphone) allows the use of assistive technologies to bridge the gap between a user with a disability and a service or target device, e.g., a TV.

[4] ONCE is the main Spanish organization for blind people. It was a partner in the INREDIS project through one of its companies, Technosite.

[5] For example, blind people interact with tactile interfaces by means of an immediate audible feedback.