Sensing, Actuation Triggering and Decision Making for Service Robots Deployed in Smart Homes

Mortaza S. Bargh[1], Melvin Isken[2], Dietwig Lowet[3], Niels Snoeck[1], Benjamin Hebgen[4], and Henk Eertink[1]

[1] Novay (NL)
[2] FASS (Gr.)
[3] Philips Research (NL)
[4] NEC Eurolabs (Gr.)
{1name.2name}@{novay.nl,offis.de,philips.com,neclab.eu}

Abstract. The Florence project develops a robot for elderly that provides multiple Ambient Assisted Living (AAL) and Lifestyle services with a consistent user-interface. The project success is measured by the acceptance of these services by the user group. Enabling such service robotics in (smart) homes requires a robust and flexible platform to collect, enhance and distribute sensory information; and to manipulate the actuators in the environment. A key characteristic of such an environment is that sensors and actuators are not always available, are distributed, and are mobile (due to e.g. the robot and phone mobility). This dynamicity requires a loose coupling between services and sensors/actuators. The paper describes the design principles and high level architecture of the Florence platform that hides the distribution, availability and mobility aspects from the services, and sketches some challenges that lie ahead.

1 Introduction

Recent demographical changes increase the demand for both professional and volunteer care in our society. The cost of care increases while the social inclusion of elderly decreases steadily. Enabling elderly to live independently as long as possible will significantly reduce the cost of care for our aging society. The Florence project is an EU FP7 project that uses robot technology to provide Ambient Assisted Living (AAL) services and to increase the level of independence of elderly. Provisioning of these services must be cost-effective, preserve the quality of life, increase the safety of elderly, and support the interaction among family-members/care-givers.

Robots are intelligent platforms that can move autonomously to interact with users, sense the environment, or perform actions. Both robots and AAL services, however, suffer from low acceptance by elderly [1]. Thus, the main challenges that the Florence project faces are cost effectiveness and user acceptance of AAL services and robots.

A key enabler of the Florence system is a platform component that makes it possible to use the sensors and actuators of the robot and those embedded in the environment (in the home, personal mobile devices, etc) by Florence-compliant services. The set of sensors and actuators available is dynamic and distributed at any time due to robot mobility, capabilities of the home infrastructure, and the load on

D. Keyson et al. (Eds.): AmI 2011, LNCS 7040, pp. 253–257, 2011.

resources. The Florence platform provides a loose and location-transparent coupling between services and sensors/actuators whereby services do not need be configured with interfaces of specific sensors/actuators. This position paper describes our design principles, the high level architecture of the Florence platform, and the challenges to be addressed in upcoming implementation phases.

2 Background and Objectives

To enhance user acceptance, the Florence project offers social and fun services next to AAL services in order to persuade users to adopt the Florence robot early on, way in advance of needing AAL services. Early adoption of the Florence robot, i.e. when users are healthy, decreases the technology adoption barrier because users perceive robots as lifestyle devices and become proud of having one. The Florence services envisioned for the initial deployment of the system are: Fall Handling (to inform caregivers if elderly fall), Keep in touch (to enable social interactions with friends and family members), Lifestyle Improvement (to coach elderly about nutrition and physical activeness), Data Logging (to collect and analyse health related data), Advanced Home Interface (to enable the remote control of home appliances and actuators), Agenda Reminder (to remind elderly of agenda events, including medication reminders) and Collaborative Gaming (to promote social interactions via gaming). This basic set of services was determined by user studies, but it is extendable with new services. Florence will use consistent interaction mechanisms to reduce learning curves, increase usability, and improve the robot adoption.

To be cost effective, the Florence project uses an off-the-shelf robot [2] to which a touch screen is added. This Florence robot must work acceptably in homes with limited sensor and actuator capabilities and be able to integrate seamlessly with a home infrastructure that possesses enhanced capabilities. A key enabler of the system is a distributed software platform that federates sensors and actuators embedded in the environment with multiple Florence services in a dynamic setting, enhances the semantics and quality (e.g., accuracy, precision) of raw sensory information gathered, and coordinates among multiple services to share resources in a user acceptable way. This position paper covers design and specification of this Florence platform.

3 Guiding Design Principles

The setting in which the Florence system operates is dynamic. Sensors and actuators appear and disappear due to robot mobility and availability (e.g. when the robot is in the docking station charging its battery one cannot rely on the robot's mobility to monitor remote locations anymore). On another scale, the sensors and actuators in home infrastructure may be dynamic in the sense that they cannot be predefined during the design phase. Nevertheless, there might be a requirement on minimum set of home devices. The Florence system must adapt its service quality gracefully when the robot or home infra is (partly) unavailable or the robot is deployed in a new home.

Supporting *graceful performance degradation* requires that application services are able to adapt to these changes in an acceptable way for elderly and there exists an abstraction layer that hides the infrastructural changes from the applications as much as possible. This paper focuses on the latter requirement and describes the so-called Sensing Actuation Decision-making Enabling (SADE) service to realise the Florence abstraction layer. The SADE service is delivered by a *software platform* component that enables a *loose coupling* of application components with sensors/actuators. To this end, we have opted for the SOA (Service Oriented Architecture) paradigm, where applications should use device-discovery services and device independent semantics for accessing sensing and actuation services. We specifically will use QoC (Quality of Context) attributes [3] for describing the sensing services. For example, instead of asking for user location obtained from a specific ultrasonic sensor, the application asks the platform for user location with one meter accuracy. Similarly, we will describe actuating actions at a high abstraction level. For example, instead of asking for turning the light on at a specific room after somehow knowing in which room the user is, the application will ask the platform for turning the light on wherever the user is (i.e., without the application directly knowing the user location).

The SADE service continuously monitors the environment, makes a high-level task plan (i.e., makes decisions), and executes the plan by actuators. For complex systems as robots, decision making is a distributed functionality that takes place at different levels of the system. We distinguish between application specific decisions that are concerned with intra-application behaviours and generic decisions that are concerned with inter-application behaviours. Generic decisions affect shared resources like the GUI screen, robot location, and robot camera direction. Application specific decision making is up to application developers and out of scope of this paper. We consider planning of generic decisions as responsibility of the platform component that must be done delicately to enhance user acceptance. Competing for processing or user-interface resources is common for multiple services running concurrently on PCs. Competing for robot location or robot camera direction is quite unique for robotics, specially for the Florence setting that deals with multiple services concurrently.

4 High Level Architecture

Figure1 illustrates a high level functional architecture of the Florence SADE service. SADE can be decomposed into three components: Decision Making (DM), Context Management Framework (CMF), and Actuator Management Framework (AMF). The CMF manages all contextual information gathered by various sensors on the robot, in the home, on the mobile phone of the user, or on a remote system outside of the Florence home. The CMF coordinates *Sensing* components that collect, wrap and pre-process raw sensory data in a form usable for other system components. The *Sensor Access* component dynamically discovers the available sensors or context-sources and binds them with context consumers (proxy based, using publish-subscribe or request-reply modes). It also supports a *Reasoning* component that enhances the semantics and quality of the collected sensory information and wraps it into a format usable within SADE. User Localisation, Activity Detection, and Health Status Detection are three main sub-components of Reasoning that are derived from the requirements of

the basic Florence services. For these components we envision to use Bayesian based reasoning. The CMF to be used in Florence is OSGi based, and originated from the EU Magnet Beyond project [4]. The CMF is designed as a distributed system that consists of one or more *Clusters*, each one located on the robot, home server, etc. Within a CMF Cluster, in turn, there is one or more so-called *Agents* which provide for local applications access to the context information of the whole CMF Cluster.

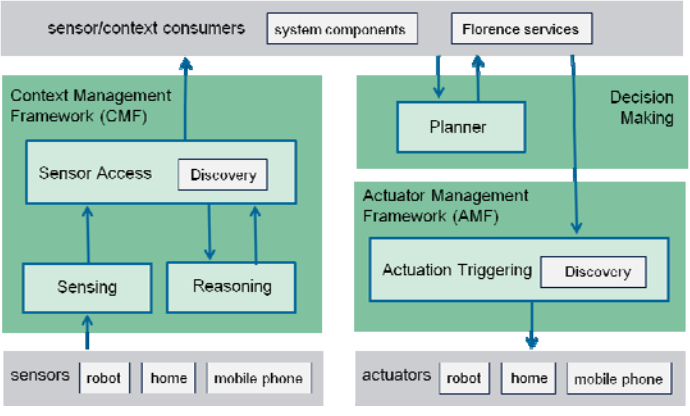

Fig. 1. High-level architecture of the Florence platform

The Florence AMF translates complex actuation tasks into elementary actuation tasks that can be executed by the actuators on the robot, home, etc. For example, a complex actuation task can be to prepare the room for doing physical exercises. The AMF will then make sure that the TV and radio are turned off, the curtains are closed, the lights are turned on, the robot is instructed to move to the appropriate position, etc. The AMF that will be developed in Florence is based on the concepts from the EU SENSEI project [5]. The AMF is distributed and uses OSGi tools to discover new actuators. Applications may use an API to request actuation tasks. In Florence we will have two types of actuation tasks: basic ones which are handled by individual actuators directly connected to the AMF (e.g., move the robot to the living room), and complex ones which are handled by composing some basic actuation tasks based on contextual information (e.g., prepare the room for doing physical exercises). This CMF-AMF architecture is similar to PERSONA's architecture described in [6].

The Florence DM component is responsible for the decision making process by decomposing high level goals into executable tasks and by prioritising and coordinating these tasks. For example, 'suggest activity X to the user' is a high level task that is decomposed to 'find the user location', 'check if the user is not busy at the moment', 'move the robot towards the user', and finally 'notify the user about the suggestion using the best available user interface'. We assume that every (application) service resolves its own internal scheduling, planning and conflicts. However, we must also ensure that application services can be developed independently. For that purpose we introduce a *Planner* component that determines the schedule of the Florence system/robot. It will prioritise competing services in a user acceptable way

by taking into account the expected schedule of the user or the urgency of a particular request (e.g., an alarm or an incoming video call may be more urgent than other services). The Planner and application services will communicate via sockets and XML-RPC. With respect to the planning algorithm, we strive for a planning that is natural, intuitive and transparent for the user. The Planner in our first implementation will be "hand-coded" to handle the initial, predefined set of services. In next steps we will investigate whether and when use of Markov Decision Processes (MDPs) and Partially Observable MDPs (POMDPs) can provide more user acceptable results.

5 Conclusion and Future Work

The requirements of the Florence system and services guided us to make an initial design of a distributed platform component for sensing, actuation triggering and decision making. The platform offers two OSGi based frameworks for sensing and actuation triggering. As the set of embedded devices is dynamic, this approach enables a loose coupling of the Florence services and the devices embedded in the environment by adopting the service discovery principle of the SOA and relying on QoC semantics to bind applications with sensors or actuators. This approach, moreover, supports graceful performance degradation of applications and eases the job of application developers. For decision making we envisioned a central Planner to determine the schedule of competing Florence services in a user acceptable way.

The architecture described in this paper serves as the initial design of the Florence platform component. Currently we are implementing the architecture outlined here by extending an existing CMF with a robot side Agent and adapting an existing AMF to the robotic setting of the Florence project. A key challenge here is to make a robust system that operates satisfactorily in a setting where the resources appear and disappear dynamically. A silent characteristic of this work is to distribute the intelligence on the robot or on the home infrastructure such that the whole system works appropriately if the robot or other resources are unavailable momentarily.

Shortly, the first user trials will start and will be evaluated using the living labs methodologies. We will rigorously examine our design principles with respect to the user acceptance criterion. This examination may lead to new requirements for planning, reasoning intelligence/learning, and definition of complex actuations.

References

1. Vastenburg, M., et al.: Designing acceptable assisted living services for elderly users. In: European Conf. on Ambient Intelligence, Nürnberg, Germany, November 19-22 (2008)
2. Pekee II robot, Wany robotics, `http://www.wanyrobotics.com/store/`
3. Sheikh, K., Wegdam, M., van Sinderen, M.: Quality-of-Context and its use for Protecting Privacy in Context Aware Systems. Journal of Software 3(3), 83–93 (2008)
4. Magnet Beyond, `http://magnet.aau.dk` (retrieved on June 10, 2011)
5. Sensei project, `http://www.sensei-project.eu` (retrieved on June 10, 2011)
6. Tazari, M.-R., et al.: The PERSONA Service Platform for AAL Spaces. In: Handbook of Ambient Intelligence and Smart Environments, p. 1171. Springer, Heidelberg (2010)