3. Detailed project description

3.1. Summary (one page)

Advances in robotics have provided solutions to problems that touch on many aspects of our daily life: industrial robots routinely support production, remotely operated robots handle dangerous tasks, autonomous robots open the way to home automation. Within the next decade, the capabilities of robots have the potential to enable spectacular new applications with critical societal benefits, such as assisted living for the elderly.

A modern robot includes both hardware and software. The hardware is the visible part, but much of the functionality of robots, and much of the potential for new applications, lies with the software. Unfortunately, the tools and techniques used to develop robotics software lag behind the state of the art in the rest of software engineering. As a result, many of the current challenges in robotics are software challenges. This situation is likely to be a major obstacle in coping with the growing complexity of new robotics applications.

Concurrency, the focus of the Roboscoop project, is one of the principal roadblocks. At the hardware level, advanced robotic systems such as autonomous mobile robots typically include many components that can in principle operate concurrently – and *should* do so to meet their potential. Sensors and actuators, for example, can run in parallel; various “arms” and “legs” can also be operated concurrently for more effective motion. Enabling such potential concurrency requires, however, that the underlying software can support concurrent execution, a goal made difficult by the limitations of traditional concurrent programming techniques, such as the standard “thread libraries” (the main concurrency mechanisms today). Programmers using such tools are constantly at risk of falling into common pitfalls such as data races (in which concurrently running units make inconsistent modifications to shared objects) and deadlocks (where computation comes to a halt because units are stuck in a circular wait on each other). More generally, concurrent applications are hard to design and implement correctly, to test and debug, and to maintain. As a result, most robotics applications limit themselves to an elementary use of concurrency, closing off many interesting possibilities offered by the hardware.

The Roboscoop project proposes to build a robust concurrency framework for robotics applications, based on rigorous theory and validated by extensive practical experience. The framework will take advantage of advances in software engineering and concurrency theory, and will be based on SCOOP (Simple Concurrent Object-Oriented Programming), a programming model for concurrency that excludes data races by construction. (Early development of SCOOP benefited from a Hasler foundation grant: DICS project #1834.)

The emerging Roboscoop framework will be closely studied and tested in a real-world setup. For the main demonstrator project we have identified Ambient Assisted Living (AAL) as an area of particular societal impact. AAL takes up the mega-trend of a continuously aging society and focuses on improving the quality of life of elderly people supported by Information and Communication Technologies.

The main showcase is the SmartWalker, a high-tech extension of the walker that many older people with reduced mobility use to move around their homes. The SmartWalker performs all the functions of today’s walkers, but is not just a metal frame with wheels: its sensors, actuators and control software bring it closer to a grandchild who helps his grandmother move around safely. We have identified several end user needs that directly result in the SmartWalker, putting the Roboscoop framework to the test and providing helpful functionality to elderly people.

The Chair of Software Engineering at ETH Zurich (ETHZ-SE) brings to the project its experience in the practice and theory of concurrency; the iHomeLab of the Hochschule Luzern (HSLU) brings extensive experience in the areas of building intelligence and intelligent living, and will apply the ideas to develop robots assisting elderly persons in their daily tasks for both comfort and security. The Autonomous Systems Lab of ETH Zurich (ETHZ-ASL), which is not requesting support as part of this project, will collaborate with the two groups by bringing its own experience in autonomous robots.

3.2. Research plan

3.2.1. Field of research

Research related to the Roboscoop project can be grouped into four main areas: programming frameworks to aid the development of robotics applications, concurrent programming languages for robotics, robots for Ambient Assisted Living, and SmartWalker-like applications.

Robotics frameworks

Several frameworks build on top of existing programming languages to ease the development of concurrent control software for robots. These frameworks provide standards, principles, applications, and libraries to support common tasks.

MOOS [Newman 08] has a layered architecture. Its *communication layer* connects *clients* (e.g. sensors, actuators, processes, etc.) through a network with a star topology. At the center of the network, there is a *server* with a database of messages. Each client bundles its messages and sends them to the server. If a client is interested in certain messages, it subscribes for messages of the right type. The client picks up the messages, whenever it connects to the server. On top of the communication layer, MOOS has an *essentials layer* with commonly used functionality such as control and logging. Applications are located on the *application layer* at the top. A *mission file* contains the configuration of the application.

ROS [Quigley 09] connects nodes in a peer-to-peer network. A node can find other nodes through a central naming service. A node sends a message by publishing it to a given topic. Other nodes can subscribe to the topic. For a single topic there might be multiple publishers and subscribers. The topic-based publish-subscribe model is not appropriate for synchronous transactions; for this purpose, ROS introduces services. A *service* offers synchronous transactions: a response follows a request. Unlike topics, only one node can advertise a service. On top of this infrastructure, ROS offers hardware abstraction, drivers, and common functionality for control, localization, planning, mapping, visual processing, logging, and simulation. To install and configure applications, ROS has an integrated package manager.

CARMEN [Montemerlo 03] has a set of modules that communicate over the network with a publish/subscribe pattern; a central hub coordinates the communication. The modules read parameters and maps from a centralized model repository. The framework follows the MVC architectural pattern. On top of that, the modules are arranged in layers. The *base layer* provides control functionality through a hardware abstraction and a set of drivers. The *navigation layer* implements localization, planning, mapping, visual processing, logging, and simulation. The *application layer* is reserved for applications. CARMEN has a separate layer for non-autonomous components, such as display modules, editors, and so on.

In LCM [Moore 03], processes exchange messages over a network through a publish/subscribe pattern. LCM does not have a central hub; the focus is on low-latency. The approach taken by LCM is to broadcast all messages to all clients. Each client discards messages to which it is not subscribed. LCM comes with a number of tools to log, playback, and inspect messages.

Player [Collett 05] is a repository that provides network-oriented programming interfaces to devices (e.g. sensors, actuators, processes, etc.). It has two layers: a core layer and a transport layer. The *core layer* deals with drivers and configurations. Each driver has an incoming message queue and can publish messages to the incoming queue of other drivers. Alternatively, a driver can also respond to requests from other drivers. The *transport layer* provides the facilities to communicate with the server and the drivers over the network.

Microsoft's Robotics Development Studio [Jackson 07] differs from most other systems: instead of a publish/subscribe model, it only uses a service model that follows the REST pattern. Messages are transmitted directly between services or over the network. The framework provides a number of predefined services to model common elements of a robotic system, such as different types of sensors and actuators; contracts describe each type of service. A discovery service lists all currently running services that conform to a certain contract. On top of that, the framework offers a simulation tool and a visual programming tool.

The summary of further frameworks can be found in [Mohamed 08]. The description of LCM [Moore 03] contains a more detailed survey covering the frameworks considered in this overview.

Concurrent programming languages for robotics

A number of concurrent programming languages have been developed for or applied to programming of robotics applications.

An early attempt to provide a concurrent programming language for robotics was Concurrent C [Cox 89], which uses rendezvous-style communication to provide synchronization on top of a general-purpose language. URBI [Baillie 05] is an object-oriented script language that is coupled with useful primitives for the parallelization of tasks and for flexible handling of events. It enables the expression of complex synchronization constraints using conditions.

Other languages for robotic control have a more domain-specific character. Frob [Peterson 99] is based on a functional language core and provides a variety of useful abstractions for programming robotic applications in a declarative way. TDL [Simmons 98] is an extension of C++ that provides explicit syntax for task-level control synchronization.

SCOOP [Meyer 97, Nienaltowski 07, Morandi 10] – which we plan to use as the basis of the Roboscoop framework – takes a middle ground: it provides a higher level of abstraction than Concurrent C through object-orientation, thus allowing for a more modular design; on the other hand it provides fewer restrictions on programmers to express their intentions than a domain-specific language.

AAL Robotics

The objective of AAL (Ambient Assisted Living) is to improve the quality of life of elderly people supported through the use of ICT. One important aspect hereby is to let the people live within their own private homes independently and safely with comfort and dignity for as long as possible. Robotics is considered a discipline of its own within AAL [AALRobotics] and also plays an increasing role in health applications. Different types of assistive robots are envisioned to bring benefit to areas such as rehabilitation, care, therapy, mobility, servicing/household, safety, wellness or social interaction. The IEEE has founded a technical committee on Rehabilitation & Assistive Robotics [IEEERobotics]. Several products are on the market today and about a dozen active projects are conducting research in the field.

A well-known example of an existing product is the social/therapeutic robot Paro [ParoRobot]. Paro is an advanced interactive robot developed by AIST that looks like a seal. It uses its sensors, actors and artificial intelligence to mimic animal behavior and develop its own character over time. The robot has several proven positive effects on patients and caregivers such as reduction of stress, stimulation of interaction, improvement of relaxation, motivation and socialization.

An example of a European research projects in AAL robotics is the FP7 project KSERA [KSERA], which aims at developing a socially assistive robot which helps elderly people, especially those with Chronic Obstructive Pulmonary Disease (COPD). KSERA also includes a connection from robot to an intelligent home environment that, among other abilities, is able to recognize falls and call for help.

The FP7 project RoboEarth [RoboEarth] will develop a World Wide Web for robots including nurse-bots and socially assistive robots. The robots use the WWW to store and retrieve learned tasks and actions. The idea is that robots learn from each other and apply the new knowledge in their own setting.

Florence [Florence] is a sister project in the EU's FP7 objective “ICT & Aging: service robotics for aging well”. It researches a multi-purpose mobile robot platform. Florence aims at delivering new kinds of AAL services to elderly persons and their caretakers. The main objective is to make robots acceptable for the users and as well as cost effective for all involved stakeholders. Florence adopts a service-oriented approach in order to support the seamless integration of capabilities provided by the robot, the home, and any required remote service providers. The project takes an off-the-shelf robotic platform, the “Pekee II” from Wany Robotics.

CompanionAble [CompanionAble] runs in the same context as Florence. CompanionAble seeks for synergies between robotics and ambient intelligence technologies and focuses on their semantic integration with the goal to provide care-givers with an intelligent assistive environment. The environment shall provide cognitive stimulation and ease the therapy management of the care-recipient. The solution incorporates a robotic companion (mobile facilitation) working collaboratively with a smart home environment (stationary facilitation). CompanionAble uses the SCITOS G5 robotic platform from MetraLabs.

Robotics projects that run in the context of the European AAL joint programme heading into similar directions are Domeo, ALIAS and ExCITE [AALBrochure 10]. A good overview over the potential of robotics for health care applications, the current research and the state of the art is also given by the EC study “Robotics for Healthcare” [Butter 08] and [Robotland 11].

Finally, a recent study [Meyer 11] (no connection) has found that a majority of both seniors and care-givers have a positive attitude towards assistive robots. They would especially accept them if they could help them to stay longer at home. Considerable effort, however, must be spent in order to improve the usability and the human-robot-interaction as many respondents still find personal robots scary and some people reject them completely. For user-friendly, reliable and safe robots the study predicts two-digit growth rates.

Smart Walkers

[Meyer 11] evaluated different applications with potential end-users and identified a particular interest for domotic robots and a robot-driven wheelchair. The SmartWalker vision of Roboscoop heads into this direction but takes a "walker" mobility aid as a base (see Figure 1).

Figure 1. Traditional walker (source: gehhilfen.org)

The idea of a smart walker has been proposed before and led to the identification of a family of problems for robotics research. Research on a smart mobility aid started in 1995 and was first mentioned in 1998 in a publication presenting the project PAM-AID [Lacey 98]. The paper described “a novel application of mobile robot technology to the construction of a mobility aid for the frail blind”. The aid would support the person walking behind it and ensure safer traveling through obstacle avoidance. The aid navigated with combination of a laser scanner (Sick LMS200) and a sonar system. Close considerations were given to the requirements for the design of such a walker applying a user-centric approach. The work resulted in two different demonstrators featuring different functionalities and user interfaces [Lacey 00]. One of the demonstrators was active (self-propelled), the other one passive (user-pushed). Both were found to be acceptable but users requested additional features such as a closer integration with the building. The PAM-AID has been tested for performance and safety [Rentschler 03] and finally renamed to Guido the Robotic SmartWalker [Rodriguez 04]. The latest development for Guido was a map based navigation system in 2005 [Rodriguez 05].

The Care-O-Bot [Schraft 98] is a universal robotic home assistant platform developed by the Fraunhofer IPA. The Care-O-Bot III has a height of 145 cm and weights 180 kg. It is self-propelled by eight motors and manipulates its environment over an arm equipped with a gripper holding up

to 3kg payload. It is packed with sensors (laser scanners, tactile sensors) and is controlled by three high performance PCs. The Care-O-Bot platform can also act as SmartWalker. The robotics engineers of IPA have mainly studied navigation and guidance problems on it [Graf 01, Hostalet 02, Graf 07, Graf 09].

Another system is the SmartCane called PAMM of the MIT [Dubowsky 99, Godding 99, Spenko 01]. The objectives of the prototype were to provide equal or better stability than a standard four-point cane, guidance to destination via pre-programmed maps, schedules, user commands and sensed obstacles, and continuous health monitoring. The relatively small and agile system used ultrasonic sensors to detect obstacles and a camera pointing at the ceiling to navigate via fixedly mounted ceiling signposts. Force and torque sensors at the handles allowed to precisely monitor the person walking behind it who could control the cane over voice commands. The SmartCane knew four different modes of operation allowing the user to control the direction in which the cane heads or letting the cane lead the way.

The robotically augmented walker presented in [Glover 03] was not only able to guide the user (using the CARMEN navigation kit) but was also able to put itself into a parking position and to find its way back to the user called over a remote control. Later developments implemented approaches that learned from the user controlling the walker and detect certain user models [Glover 04].

Recent works on Smart Walkers include three projects with the name “iWalker” [Kulyukin 08, Annicchiaricov 08, Röfer 09]; the Rolling Walkers [Cornell 10], featuring patented electronic brakes; and works focusing on activity monitoring and intention based control, applying artificial intelligence and machine learning algorithms [Omar 10].

3.2.2. Goals and objectives of the project

The aim of the Roboscoop project is to bring a new degree of power and flexibility to robotics by enabling the use of full-fledged concurrency (parallelism) to robot programming. As previewed above, the showcase applications run on a SmartWalker, a robot for assisted living services for senior citizens, to be presented in the iHomeLab.

The project will develop both a general *Roboscoop programming framework* and *Roboscoop applications* utilizing the framework.

The programming framework is based on SCOOP (Simple Concurrent Object-Oriented Programming) [Meyer 97, Nienaltowski 07, Morandi 10], a practical programming model for the development of high-quality concurrent software that carries the advantages of object technology to the concurrent context. Section 3.2.3 will present both the SCOOP model and its previous robotics case study.

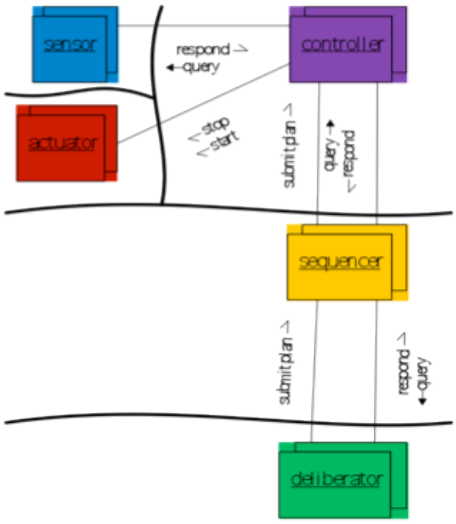


Figure 2. 3-layer architecture of the Roboscoop framework

We expect that the Roboscoop framework will use a three-layer architecture [Gat 97] as outlined in Figure 2:

- The *control* layer implements one or more stateless feedback control loops that couple sensors to actuators. This layer includes all of the control software; it enables primitive robot behaviors.
- The *sequencer* layer interacts with the control layer to fulfill a task; it is stateful and relies on primitive behaviors from the control layer.
- The *deliberator* layer performs time-consuming deliberative computations, producing plans for the sequencer layer.

We expect the framework to use SCOOP on all three layers; the framework will manage inter-layer access.

The applications will be based on the framework and pursue two main objectives:

- Demonstrating the functionality, usability and applicability of Roboscoop by identifying and addressing the technical challenges related to common robotics problems and solving them elegantly.
- Solving real-world problems in an area of crucial societal impact: Ambient Assisted Living (AAL) robots for the elderly. The SmartWalker will be the principal testbed for this part of the project.

The SmartWalker robot of the project will be self-propelled and equipped with various sensors that, depending on the application, allow it to:

- Move around together with the user (either by actively taking him along or more passively helping him to get to a desired location).
- Move around independently.
- Identify its own position, both indoors and outdoors.
- Navigate safely, both indoors and outdoors.
- Interact with the user through different means such as touch display, natural language, and other sensors/actuators.
- Monitor the user's condition.
- Connect to a home network and the Internet.
- Be remotely controlled.

This combination of mechanisms makes the SmartWalker a highly concurrent device and an excellent testbed for a fully concurrent robotics programming framework. Clearly not all the listed applications can be implemented as part of the project; we will have to select a subset for implementation. Criteria for this selection are given below.

In terms of the possible applications, several ideas, some relatively straightforward and others more speculative, could be envisioned for Roboscoop:

- Monitor the user's gait, heart rate and other parameters to evaluate his health and physical condition.
- Support the user while shopping (walkers often come with a basket and are used as shopping carts).
- Let the walker return to a parking or docking station automatically and get back to the user on its own if needed.
- Provide automated support for folding and unfolding the walker.
- Let the walker support care-givers and relatives directly in
    - detecting emergencies such as falls or inactivity,
    - sounding an alarm in case of an emergency,

    - letting the care-givers access and control it remotely including a live streaming image that lets them check the situation at home.
- Lead the user along walker-qualified walking trails or even provide a kind of “guided tour” and make sure he arrives safely. During the walking, support the user with
    - the force to move the device itself, especially uphill,
    - an automatic braking mechanism, so that the walker does not roll away by accident.
- Guide the user to a specific indoor target site, e.g. an examination room in a hospital.
- Use the walker as a specialized physical training device at home including a kind of “personal trainer” giving motivational tips.

The SmartWalker is the showcase application for the Roboscoop project, and the most challenging, but we intend to apply the framework to other, smaller developments to ensure generality and application-independence. In particular, we will use the framework for teaching at ETH Zurich, where several smaller example applications will be developed as part of student projects and/or course projects using Lego Mindstorms robotic kits. For this educational component of the project we will initially use the CCC course, Concepts of Concurrent Computation, which the Chair of Software Engineering has taught every year since 2005 to an average of about 25 students.

3.2.3. Scientific methods to be applied

The Roboscoop project will be founded on earlier scientific contributions of the Chair of Software Engineering, in particular the SCOOP [Meyer 97, Nienaltowski 07, Morandi 10] programming model. (The SCOOP compiler is part of the EiffelStudio development environment and can be downloaded as open source [EiffelStudio]). The project also relies on the chair’s initial foray into robotics, which led to a paper presented at a major international robotics software conference [Ramanathan 10]. We will describe these contributions first and then outline methods for collecting requirements and building applications for the SmartWalker showcase.

The SCOOP programming model

The SCOOP model provides a structured approach to developing programs that require concurrency. It extends familiar notions from object-oriented programming and Design by Contract to concurrent behavior. Traditionally, programmers needing to add concurrency to their programs have had to use low-level mechanisms such as mutexes and semaphores, introducing a conceptual gap between the behavior of a program and its implementation. SCOOP narrows this gap.

The Chair of Software Engineering has performed initial developments towards the Roboscoop project by applying SCOOP to a number of robot projects in recent years. The most recent and most significant is the Hexapod project [Ramanathan 10]. A movie showing the Hexapod robot in action is available at http://tinyurl.com/scoop-robotics. Figure 3 is a photograph of the robot.

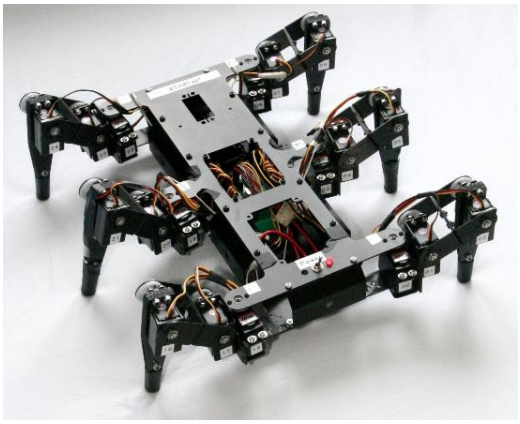


Figure 3. The Hexapod robot

The Hexapod robot emulates the gaits (leg movement patterns) of hexapods in nature. In tripod gait, two sets of legs execute alternating sequences of movement (see Figure 4): the first group lifts the legs off the ground and swings forward and places the feet on the ground again; the second group then lifts and swings forward, concurrently with the first set, which now moves the legs backward while being placed on the ground, thus propelling the body forward. This cycle is repeated continuously.

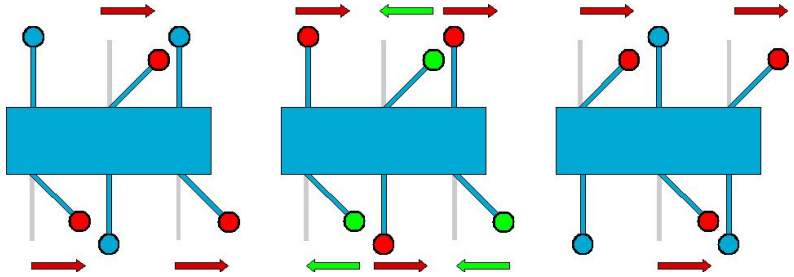


Figure 4. Tripod gait

To preserve stability while walking, a hexapod must allow lifting a group of legs off the ground and moving them to the front only if (REQ1) they are already retracted (moved to the back) and (REQ2) the partner group is firmly planted on the ground. Such rules, which any implementation must observe, are part of the system's specification.

With SCOOP, these requirements can be explicitly stated as *wait conditions* of a routine using the require keyword, meaning that the routine execution is delayed until the conditions hold.

The following routine can be used to implement the leg-lifting phase:

```
begin_protraction (partner_signaler, my_signaler: separate SIGNALER)
  require
      my_signaler.legs_retracted        – – REQ1
      partner_signaler.legs_down        – – REQ2
  do
      legs.lift
  end
```

When this routine is called, the SCOOP runtime system waits until it can lock objects *partner_signaler* and *my_signaler*. When this happens, no other thread can access these objects. The wait conditions here directly satisfy the REQ1 and REQ2 specifications. Expressing synchronization requirements in this way provides a natural framework for implementing coordination requirements in robotics applications.

SmartWalker Requirements

The Roboscoop project, will build an open SmartWalker-platform. As noted, not all potentially interesting applications, from the ones listed above, can be implemented as part of the present project. The main criteria are:

- The scientific relevance of the robotics challenges an application raises.
- Its potential value for users.

End-user involvement is critical for the success of any AAL application. Therefore, elderly users shall be involved in the development process at all levels, in particular during requirements gathering and during the trials of the technology. A user-driven process will be applied, including open interviews, focus groups and user trials.

- *Open Interviews* Visit two or three existing walker users and let them talk about their walker and how they use it. The open interview concentrates on a few persons representing the target group and lets them talk quite freely without fixed questionnaires. The statements and findings are collected in detail. A concluding workshop enables the development team to derive similarities, directions and concrete requirements from the interviews.
- *Focus Groups* Once a first draft of the requirements has been created and a rough picture of the "product" is ready, collect a number of potential customers from the target segment. Present the idea to them, including services and pricing, and receive qualitative feedback on the offering. This will be used to adapt the offering in a first step to customer needs.
- *User Trials* Use 3 to 5 existing walker users. Closely follow and analyze their usage of the new device, get active feedback on usability, added value to their daily lives, and willingness to pay for such new services.

SmartWalker Platform

As for the hardware of the SmartWalker, we will take an existing walker and use it as a base for extensions and modifications in order to "make it smart". We will rely on the advice of the ETHZ-ASL to evaluate, select and adapt standard COTS (Commercial Off-The-Shelf) sensors, actors and batteries as well as an embedded PC-based control platform.

It is not the aim of the Roboscoop project to come up with new robotics-specific algorithms, for such tasks as navigation or obstacle avoidance. For these goals we will extensively rely on existing, published mechanisms, on expertise from ETHZ-ASL, and on standard, open-source libraries and components.

3.2.4. Significance of the planned research

The two principal contributions of the Roboscoop project are:

- To bring a new level of power and flexibility to robotics, by enabling the development of advanced robotics applications in many areas, taking advantage of a high level of concurrency.
- To demonstrate the applicability of the concepts through the construction of robots in an area of critical societal benefit, Ambient Assisted Living for the elderly.

On the software engineering side, we expect new contributions to the theory and practice of concurrent programming. On the application programming side, new insights and guidelines on programming concurrent robot applications are expected as seen from the perspective of an application developer.

From the perspective of AAL and the end-users, the contribution is a SmartWalker idea mature enough to motivate industrial partners to take its development a step further into the direction of a product that not only brings new advances to device and service providers but, most importantly, dramatically eases the life quality of elderly people and eases the burden on their care-givers.

3.3. Project planning, time scheduling, and resource allocation

3.3.1. Work packages and resource allocation

The following resources are requested as part of the Roboscoop project. The exact figures are detailed in the Project Funding section of the Proposal Form.

Personnel, for the 3-year duration of the project:

- One PhD student at ETHZ-SE
- One researcher (wiss. Mitarbeiter) at HSLU
- Hardware, utilizing commercially available walker and robotics components to build two SmartWalker demonstrators.
- Commercially available robotics kits (Lego Mindstorms) to use for teaching at ETH and to build smaller example applications for the framework testing.

- Travel support for presentations at computer science, robotics, and AAL conferences.

The project allocates the resources within the work packages outlined below.

WP1: Framework review and requirements

Duration: 6 months

Personnel resources: ETHZ-SE, ETHZ-ASL (advisor)

Description:

As an initial step this work package includes a review of existing middleware frameworks, conceptual architectures, and applications (as outlined partly in Section 3.2.1). Based on the review of existing applications, we will select a number of basic use cases that allow us to define the requirements for the framework. The use cases will include SmartWalker applications, making use of the information gathered in WP3. Furthermore, a number of smaller robotics applications shall be defined, realizable in the Lego Mindstorms framework. Given the reviews and use cases the framework requirements will be defined.

WP2: Framework design and implementation

Duration: 24 months

Personnel resources: ETHZ-SE

Hardware resources: Robotics kits (Lego Mindstorms)

Description:

Based on the findings of WP1, we will define a suitable framework architecture (possibly a three-layer architecture, as outlined in Section 3.2.2) and an interface to the middleware. The SCOOP mechanism will be used as a language for concurrency control, and also to implement (parts of) the framework itself. We intend to release an early prototype implementation of the framework with limited feature set as part of milestone M1. The prototype will be assessed by using it for teaching at ETH Zurich and by applying it to Lego Mindstorms use cases, defined as part of WP1. Based on the lessons drawn from the assessment, the design and implementation will be completed in a Version 1 release. The implementation will be finalized by feedback obtained from programming SmartWalker applications (WP4) and the preparation of the iHomeLab showcase (WP5).

WP3: SmartWalker platform requirements

Duration: 9 months

Personnel resources: HSLU, ETHZ-ASL (advisor)

Description:

Applying a user-driven innovation process, the end user requirements for the SmartWalker are gathered in Open Interviews, Workshops and Focus Groups. Taking the end-user needs, a set of concrete use case scenarios is identified and documented. These scenarios shall be implemented and tested on the SmartWalker in WP4. Derived from these scenarios, the technical requirements for the hardware- and software-platforms are defined and written down. These include the functionalities and interfaces which the framework needs to provide to the application.

WP4: SmartWalker platform design and implementation

Duration: 21 months

Personnel resources: HSLU

Hardware resources: SmartWalker hardware

Description:

In this work package the hardware and software of the SmartWalker will designed, implemented and tested. First, the SmartWalker hardware needs to be designed and the components evaluated. In the next step the SmartWalker will be built by taking the evaluated walker frame and extend it with the additional components needed to make it smart (and also look smart if possible). In the next step, the framework needs to be ported to the new hardware, which includes the development of an easy to use SmartWalker API interfacing all the sensors and actuators as well as a graphical

user interface if needed. Finally, the application scenarios are implemented and tested, involving developers and end-users.

WP5: iHomeLab Showcase

Duration: 12 months

Participating partners: HSLU, ETHZ-SE

Description:

The results of Roboscoop, including the SmartWalker, shall be presented to the public during the official guided tours in the iHomeLab. This poses special requirements on a demonstrator. First, it has to transport a clear message in telling a simple story that people can follow. We want to give them an impressive user-experience, something to remember after leaving the iHomeLab. Second, its another quality of application that is needed here. It does not need to have the characteristics of a finished product. But an iHomeLab-Showcase is certainly more mature than research prototype that works in a laboratory. This means it should look appealing and work 100% reliably and be reproducible at any time.

3.3.2. Deliverables

The project will build two SmartWalker robots for public demonstrations, and use the programming framework for teaching at ETH Zurich. We expect publications in such venues as IROS (IEEE Intl. Conference on Intelligent Robots and Systems, where the initial hexapod paper [Ramanathan 10] was published), ICRA (IEEE Intl. Conference on Robotics and Automation), or JOSER (Journal of Software Engineering for Robotics). Further, the results of the Roboscoop project will be published and presented at reputed European AAL conferences, namely the AAL-Forum, the yearly main event of the AAL research community, and the German AAL-Congress.

All software developed by the Roboscoop project will be made available in open-source form. The hardware and software of the SmartWalker may be promoted as the basis for an open platform for AAL walker applications.

In particular, the following deliverables will be produced as part of the project results:

D1: Framework design and early prototype (ETHZ-SE)

D2: SmartWalker requirements and platform evaluation (HSLU)

D3: Version 1 of framework implementation (ETHZ-SE)

D4: SmartWalker system design and prototype implementation (HSLU)

D5: Final version of framework (ETHZ-SE)

D6: SmartWalker final implementation and integration into iHomeLab (HSLU)

3.3.3. Milestones

To measure the progress of the project and to decide on potential adjustments in the work schedule, yearly milestones are set.

Milestone M1

When: Month 12

Deliverables:

- **D1** Framework design and early prototype (ETHZ-SE)
- **D2** SmartWalker requirements and platform evaluation (HSLU)

Milestone M2

When: Month 24

Deliverables:

- **D3** Version 1 of framework implementation (ETHZ-SE)
- **D4** SmartWalker system design and prototype implementation (HSLU)

Milestone M3

When: Month 36

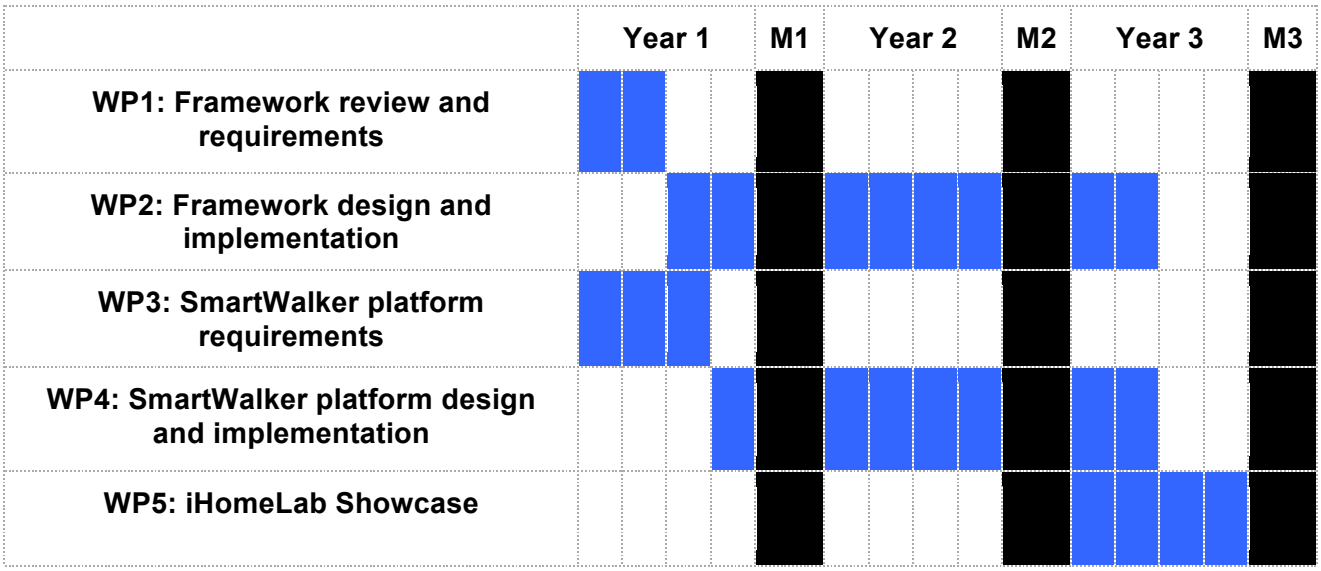
Deliverables:

- **D5** Final version of framework (ETHZ-SE)
- **D6** SmartWalker final implementation and integration into iHomeLab (HSLU)

3.3.4. Time schedule

The Roboscoop project is planned for a duration of three years, with a start date of 1 October 2011 and an end date of 30 September 2014.

The following diagram gives an overview of the scheduling project's work packages (detailed above in Section 3.3.1) and milestones (detailed above in Section 3.3.2).



References

[AALBrochure 10] AAL-JP: Overview of funded and running projects. http://www.aal-europe.eu/projects/aal-brochure-2010

[AALRobotics] AAL Forum Group: Robotics in AAL site. http://www.aalforum.eu/group/robotics

[Alavi 10] Alavi M., *SCOOP in Practice*, Technical Report 717, ETH Zurich, January 2010.

[Annicchiaricov 08] Annicchiaricov R., Barrué C., Benedico T., Campana F., Cortés U., Martínez-Velasco A.: *The i-Walker: an intelligent pedestrian mobility aid.* In: Proceedings of 18th European Conference on Artificial Intelligence (ECCAI'08). pp-708712. 2008.

[Baillie 05] Baillie, J.-C.: *URBI: Towards a Universal Robotic Low-Level Programming Language*. In: IROS'05, 2005

[Butter 08] Butter, M. et. al.: *Robotics for Healthcare*. Final Report to the European Commission Information Society, 2008

[Collett 05] Collett, T. H., MacDonald, B. A., Gerkey, B.: *Player 2.0: Toward a Practical Robot Programming Framework*. In: Proceedings of the Australasian Conference on Robotics and Automation. 2005

[Cornell 10] *Newly patented electronic braking prevents slips on walkers*, Cornell Chronicle June 24, 2010, http://www.news.cornell.edu/stories/June10/SmartWalkers.html

[CompanionAble] CompanionAble, Integrated cognitive assistive & domotic companion robotic systems for ability & security project site: http://www.companionable.net/

[Cox 89] Cox, I. J., Gehani, N. H.: *Concurrent programming and robotics*. International Journal of Robotics Research 8:3-16, 1989.

[Dubowsky 99] Dubowsky S. *Personal Aid for Mobility and Monitoring: A Helping Hand for the Elderly.* In: Home Automation and Health Care Consortium Report, 1999

[EiffelStudio] EiffelStudio (SCOOP) download site. http://sourceforge.net/projects/eiffelstudio/

[Gat 97] Gat, E.: *On Three-Layer Architectures*. In: Artificial Intelligence and Mobile Robots, pp. 195-210, 1997.

[Florence] Multi Purpose Mobile Robot for AAL project site: http://www.florence-project.eu/

[Glover 03] Glover J., Holstius D., Manojlovich M., Montgomery K., Powers A. , Wu J., Kiesler S., Matthews J., and Thrun S.: *A robotically augmented walker for older adults.* Technical Report CMU-CS-03-170, Carnegie Mellon University, Computer Science Department, Pittsburgh, PA, 2003

[Glover 04] Glover, J.; Thrun, S.; Matthews, J.T. *Learning user models of mobility-related activities through instrumented walking aids*. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA '04), New Orleans, LA, USA, April 26–May 1, pp. 3306–3312, 2004

[Godding 99] Godding S.J.: *Field Tests on a Personal Mobility Aid for the Elderly*. B.Sc. Thesis Department of Mechanical Engineering, Massachusetts Institute Technology

[Graf 01] Graf, B.: *Reactive Navigation of an Intelligent Robotic Walking Aid.* In: Proceedings of ROMAN-2001, pp. 353-358, 2001

[Graf 07] Graf, B.; Schraft, R.D.: *Behavior-based Path Modification for Shared Control of Robotic Walking Aids*. In: 10th International Conference on Rehabilitation Robotics, June 12 - June 15, 2007, Noordwijk, Netherlands. Piscataway, NJ: IEEE Operations Center, S. 317-322, 2007

[Graf 09] Graf, B.: *An Adaptive Guidance System for Robotic Walking Aids*. In: Journal of Computing and Information Technology - CIT 17, Nr. 1, S. 109-120, 2009

[Hostalet 02] Hostalet Wandosell, J. M., Graf, B.: *Non-Holonomic Navigation System of a Walking-Aid Robot*. In: Proceedings of IEEE ROMAN '02, pp. 518-523, 2002

[IEEERobotics] IEEE Robotics and Automation's Technical Committee on Rehabilitation & Assistive Robotics. http://tab.ieee-ras.org/committeeinfo.php?tcid=18

[Jackson 07] Jackson, J.: *Microsoft robotics studio: A technical introduction.* IEEE Robotics & Automation Magazine 14:82-87, 2007.

[KSERA] Knowledgeable Service Robots for Aging project site. http://ksera.ieis.tue.nl/

[Newman 08] Newman, P. M.: *MOOS: Mission Orientated Operating Suite.* Department of Ocean Engineering, MIT, 2008.

[Kulyukin 08] Kulyukin V., Kutiyanawala, A., LoPresti E., Matthews J., Simpson R.; *iWalker: Toward a Rollator-Mounted Wayfinding System for the Elderly*, In: Proceedings of 2008 IEEE International Conference on RFID The Venetian, Las Vegas, Nevada, USA April 16-17, 2008

[Lacey 98] Lacey G., Dawson-Howe K. M.: *The application of robotics to a mobility aid for the elderly blind*. In: Robotics and Autonomous Systems 23, Elsevier Science B.V., pp 245-252, 1998

[Lacey 00] Lacey G., MacNamara S.: *User involvement in the design and evaluation of a smart mobility aid*. In: Journal of Rehabilitation Research and Development Vol. 37 No. 6, Nov/Dec 2000 pp 709—723, 2000

[Meyer 11] Meyer S.: *Mein Freund der Roboter*, BMBF/VDE Innovationspartnerschaft AAL

[Meyer 93] Meyer, B. *Systematic Concurrent Object-Oriented Programming*, in *Comm. ACM,* 36(9):56-80.

[Meyer 97] Meyer B., *Object-Oriented Software* Construction, Second Edition, Prentice-Hall, 1997

[Mohamed 08] Mohamed, N., Al-Jaroodi, J., Jawhar, I.: *Middleware for Robotics: A Survey*. In: RAM'08, 2008

[Montemerlo 03] Montemerlo, M., Roy, N., Thrun, S.: *Perspectives on Standardization in Mobile Robot Programming: The Carnegie Mellon Navigation (CARMEN) Toolkit*. In: IROS'03, pp. 2436-2441, 2003.

[Moore 03] Moore, D., Olson, E., Huang, A.: *Lightweight Communications and Marshalling for Low-Latency Interprocess Communication.* Computer Science and Artificial Intelligence Laboratory, MIT, 2003.

[Morandi 10] Morandi B., Bauer S., Meyer B., *SCOOP – A Contract-Based Concurrent Object-Oriented Programming Model*, in LNCS 6029, LASER Summer School 2008, Springer, 2010, 41-90.

[Morandi 11] Morandi B., Nanz S., Meyer B., *A Comprehensive Operational Semantics of the SCOOP Programming Model*, Technical Report, http://arxiv.org/abs/1101.1038, 2011.

[Nanz 11] Nanz S., Torshizi F., Pedroni M., Meyer B., *Empirical Assessment of Languages for Teaching Concurrency: Methodology and Application*, in CSEE&T 2011, IEEE Computer Society, 2011. To appear.

[Nienaltowski 07] Nienaltowski P.: *Practical framework for contract-based concurrent object-oriented programming*, PhD dissertation 17061, Department of Computer Science, ETH Zurich, February 2007.

[Nienaltowski 09] Nienaltowski P., Meyer B., Ostroff J.S., *Contracts for concurrency*. In: Formal Aspects of Computing Journal 21(4): 305-318, 2009.

[Omar 10] Omar, F., Sinn, M., Truszkowski, J., Poupart, P., Tung, J., Caine, A. *Comparative Analysis of Probabilistic Models for Activity Recognition with an Instrumented Walker*. In: Proceedings of the 26th Conference on Uncertainty in Artifical Intelligence (UAI), 2010

[ParoRobot] Paro Therapeutic Robot site. http://www.parorobots.com/

[Peterson 99] Peterson, J., Hager, G. D., Hudak, P.: *A Language for Declarative Robotic Programming.* In: ICRA'99, 1999.

[Quigley 09] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T. B., Leibs, J., et al.: *ROS: an open-source Robot Operating System*. In: ICRA Workshop on Open Source Software, 2009

[Ramanathan 10] Ramanathan G., Morandi B., West S., Nanz S., Meyer B., *Deriving Concurrent Control Software from Behavioral Specifications*, In: Intelligent Robots and Systems (IROS'10), IEEE, 2010.

[Rentschler 03] Rentschler J., Cooper R. A., Blasch B., Boninger M. L.: *Intelligent walkers for the elderly: Performance and safety testing of VA-PAMAID robotic walker*, In: Journal of Rehabilitation Research and Development Vol. 40, No. 5, Sep./Oct. 2003 pp. 423–432, 2003

[RoboEarth] RoboEarth project site: http://www.roboearth.org/

[Robotland 11] Ambient Assisted Robotics in Europe. http://robotland.blogspot.com/2011/01/ambient-assisted-robotics-in-europe.html

[Rodriguez 04] Rodriguez-Losada D., Matia F., Jimenz A., Galan R., Lacey G.: *Guido, the Robotic SmartWalker for the frail visually impaired*. In: Proceedings of First International Congress on Domotics, Robotics and Remote Assistance for All, 2005

[Rodriguez 05] Rodriguez-Losada D., Matia F., Jimenz A., Galan R., Lacey G.: *Implementing Map Based Navigation in Guido, the Robotic SmartWalker*. In: IEEE International Conference on Robotics and Automation, Barcelona, Spain, pp3401 – 3406, 2005

[Röfer 09] T. Röfer, T. Laue, B. Gersdorf, *iWalker - An Intelligent Walker providing Services for the Elderly*, in Technically Assisted Rehabilitation, 2009

[Schraft 98] Schraft R.D., Schaeffer C. and May T., *Care-Obot(tm): The Concept of a System for Assisting Elderly or Disabled Persons in Home Environments*, In: IECON: Proc. of the IEEE 24th Annual Conf., V.4, pp.2476-2481, 1998

[Simmons 98] Simmons, R., Apfelbaum, D.: *A task description language for robot control*. In: IROS'98, 1998.

[Spenko 01] Spenko, M., *Design and analysis of the SmartWalker, a mobility aid for the elderly*, Master's thesis, Department of Mechanical Engineering, MIT, Cambridge, MA, 2001.

[West 10] West S., Nanz S., Meyer B., *A Modular Scheme for Deadlock Prevention in an Object-Oriented Programming Model*, in Formal Engineering Methods (ICFEM'10), LNCS 6447, Springer, 2010, 597–612.