Int J Soc Robot (2014) 6:121–139
DOI 10.1007/s12369-013-0209-8

A Modeling Framework for User-Driven Iterative Design of Autonomous Systems

Manja Lohse · Frederic Siepmann · Sven Wachsmuth

Accepted: 19 July 2013 / Published online: 6 September 2013

Abstract Many researchers in human-robot interaction have acknowledged the fact that iterative design is necessary to optimize the robots for the interaction with the users. However, few iterative user studies have been reported. We believe that one reason for this is that setting up systems for iterative studies is cumbersome because the system architectures do not support iterative design. In the paper, we address this problem by interlinking usability research with system development. In a first user study, we identify requirements and concepts for a new framework that eases the employment of autonomous robots in the iterative design process. With a second user study we show how robot behaviors are implemented in the new framework and how it enables the developer to efficiently make changes to these behaviors.

Keywords Human-robot interaction · System architecture · Autonomous systems · Tasks · Iterative system design · User studies

1 Introduction

Evaluation is essential in the process of designing any kind of usable systems. Therefore, in the field of human-robot interaction (HRI) methods to evaluate robots have been adapted from other disciplines (e.g., interaction analysis [6, 26], task analysis [1, 24, 35], and heuristic evaluation [8]). Also measuring tools to assess HRI in diverse situations and experiments have been proposed [3, 15, 16, 18, 39]. Evaluating the systems has led to valuable insight about the robots and the way people interact with them. However, to our knowledge and as has also been noted by other authors (e.g., [1, 27]) very few iterative studies have been conducted in HRI. When looking at the proceedings of the last three HRI conferences (2011–2013), we found that only two papers out of 96 dealt with iterative design or long-term interaction [19, 22]. At the Ro-Man conferences between 2010 and 2012, 5 out of 368 papers concerned long-term interaction [28–30, 33, 37]. In their comprehensive survey of long-term studies in HRI, Leite et al. [23] conclude that research is picking up the topic, however, the studies presented so far are mostly exploratory. But iterative design is necessary to optimize the robots and their behaviors and many researchers in HRI have recognized this (e.g., [35, 38]).

We developed the Bonsai framework to address this issue particularly for HRI with autonomous systems. In Sect. 2 we describe challenges of iterative system design for such autonomous robots. We describe how we derived the conceptual foundations of the framework from findings of user studies (Sect. 3). We then introduce the Bonsai framework and explain how it supports iterative design and enables behavior transfer between robots and scenarios (Sect. 4). Thereafter, we demonstrate how Bonsai was implemented in the robot BIRON (Sect. 5) and present a short usability study that shows how well developers can handle Bonsai (Sect. 6). Finally, in Sect. 7 a case study illustrates how the framework supports efficient iterative behaviour design by taking findings from user studies into account.

M. Lohse (✉)
University of Twente, Enschede, Netherlands
e-mail: m.lohse@utwente.nl

F. Siepmann · S. Wachsmuth
Bielefeld University, Bielefeld, Germany

F. Siepmann
e-mail: fsiepman@techfak.uni-bielefeld.de

S. Wachsmuth
e-mail: swachsmu@techfak.uni-bielefeld.de

2 Challenges of Iterative Design in HRI

We encountered possible reasons for the lack of iterative approaches when we worked towards conducting iterative user studies with an autonomous robot ourselves. Our goal was to improve the autonomous system over time based on what we learned about and from the interaction. While pursuing this goal, we encountered several challenges.

The *first challenge* was a methodological one given that the robot acts autonomously. Thus, the robot behavior is not pre-scripted. In other words, based on the interpretation of the sensor input and its internal states, the system decides what actions it takes at a given point of time. Hence, no interaction is exactly like another one and we had to answer the question of how to compare robot performance between the trials. Our solution to this challenge was to evaluate the robot on the level of specific tasks. More information about this is provided in Sect. 3.

The *second challenge* that we encountered was the question of how the robot system needs to be designed in order to enable researchers to integrate findings from user studies easily and efficiently? This question was anything else but trivial because seemingly simple improvements of the interactive behavior of the robot often requires to make changes in different components of the system that have interdependencies with each other. Thus, the engineer needs a deep understanding of components, middleware, and architecture in order to make changes to the robot behavior.

Robotic systems typically are a result of a collaborative engineering process in an environment of rapidly changing technologies. They consist of a large number of hardware and software components solving problems from different research areas (navigation, mapping, perception, planning, speech understanding, dialog, etc.). Therefore, most robotic software frameworks such as ROS, CLARAty, YARP, XCF, or Player/Stage focus on hardware abstraction, the re-use of components, and middleware solutions. These frameworks improved the engineering process of robotic systems significantly. However, existing frameworks for robot development such as YARP [10] or ROS [32] have not solved the problem: Implementing robot behaviors with many software components into one system appears to be still more of an art than a systematic engineering process.

The *third challenge* that we encountered connected to the iterative design process was the question of how it would be possible to easily re-use robot behaviors that have been evaluated in other scenarios, on other platforms, or with different frameworks. This question is highly relevant because user studies and the design of robot behaviors take a lot of time and effort. Thus, it would be a great advantage if we were able to re-use robot behaviors that have been proven to work well. Reusability would ease the process of setting up systems for future iterative studies and would allow for benchmarking between systems.

Reusability has also been one aspect envisioned by Glas et al. [12]. The authors, quite related to our work, proposed an interaction design framework for social robots. However, their focus was on providing non-programmers with a graphical interface to compose sequences of behaviors. The users did not need knowledge about the robot driver layer and the information processing layer for behavior design. Even though their approach helps the goal of having non-programmers develop robot behaviors, it makes changes of the robot's behaviors laborious because they have to be adapted at all relevant parts of the sequence. Moreover, working with the graphical interface limits the developers, e.g., to using predefined blocks. We aim to avoid such issues in our approach and want to provide reusable building blocks of behavior that can be improved over time.

Connected to reusability is the *fourth challenge*: enabling easy integration of the system in different software frameworks and platforms. Hence, the framework needs to be platform-independent in order to ensure that different systems can indeed profit from evaluations of other systems and cross-system evaluations can be conducted in an efficient manner.

In conclusion, new approaches are needed which enable easy and efficient iterative design and system development in robotics. The system shall support the design of robot behaviors that can easily be re-used and adapted based on user studies. It shall combine the strengths of robotics and usability research in order to produce more capable and more usable systems.

3 Conceptual Foundations

To address the challenges mentioned in Sect. 2, we first conducted one time evaluations to develop concepts to analyze the complex interaction structure which also support iterative design. One of these concepts is the structuring of the interaction into tasks [26]. Tasks are recurring patterns on the *interaction level* (the level on which the interplay between the users and the robot can be analyzed). Such patterns form the structure of the interaction. Tasks enable the researcher to compare different interactions with each other. Each task is based on a prototypical interaction script. This script describes the combination of behaviors that the users and the robot have to perform in order to complete the task (for examples see Sects. 7.1.4 and 7.2.4). In previous work we distinguished three kinds of tasks [25]:

- social tasks
- functional tasks
- problem-related tasks

The social tasks frame the interaction and are mainly used to manage the participant's attention and the turn taking. Examples for social tasks are greetings, introductions,

and farewells. These occur in most interactions and, thus, most interactive robots need to be able to handle social tasks.

The functional tasks constitute the main part of the interaction. They are mandatory to reach the goal of the interaction and strongly depend on the interaction scenario. A typical functional task for a mobile robot is to drive to another location.

The third group subsumes the problem-related tasks. These are initiated by the robot or the user if they detect a problem in the interaction and dispose of a strategy to solve it. Most problem-related tasks are strongly connected to the sensory perception of the system. Typical examples for problem-related tasks are the robot getting stuck at obstacles and asking for help or the robot not detecting the user anymore and trying to detect him/her again (see Case Study in Sect. 7). The problem-related tasks enable the robot and the user to continue the interaction even if there is a problem.

Analyzing the problem solving procedures as tasks allows us to represent them in a transition matrix with all other tasks in order to determine when they occur. Thus, all tasks are represented in a task structure. By task structure we refer to the quantity and order of the tasks. It offers important insights in the course of the interaction, e.g., which tasks cause most problem-related tasks or which tasks are most commonly initiated by the users. Typically in HRI, only one task occurs at a time and the task structure is linear.

However, the task structure alone does not tell whether the goals of the tasks were achieved. Therefore, also failures within tasks (deviation patterns for each prototypical interaction script) need to be analyzed. To give an example of such a failure: guiding the robot might not end in a problem-related task (e.g., the robot recognizing an obstacle and asking for help) but, nevertheless, the robot might stop before the user asked it to do so, e.g., because it mistook another utterance for a stop command. Thus, the robot could not detect a problem (as it would have needed to initiate a problem-related task). This example points to the fact that failures that are apparent on the interaction level often have their cause on the *system level*. The system level describes the robot components and their interplay.

To analyze both interaction level and system level at the same time we have previously proposed the SInA (Systemic Interaction Analysis) approach [26]. SInA allows for a careful description of the interaction and helps to determine the relations between the user's and the system's behavior. SInA is used to identify what the robot does, what happens within the robot, and what the users do. Deviations from the prototypical interaction scripts and their causes (i.e., inappropriate expectations of the user or inadequate design of the robot) can be identified.

Typical deviations from prototypical interaction scripts that we found in previous studies were problems with speech understanding, person perception, the robot's states (one action was not finished and a new one could not be started), and navigation (the robot getting stuck at an obstacle or the user standing within the security distance of the robot). We proposed that changing either the robot feedback, the components, or the system architecture could solve each of these problems [24, 26]. To give some examples: a change to the robot feedback could make the robot ask the users to step back, the users would get out of the robot's security distance, and the robot could start driving; an improvement to the speech recognition component could decrease the number of speech understanding problems; and changes to the architecture could allow participants to start new actions before previous ones are finished.

This view was strongly connected to the traditional system design that required making changes at different levels of the system and often within various components. However, as has been argued above, this often makes iterative design a very tough challenge. Thus, we concluded that adapting the system based on findings from user studies would be much easier if we could make changes on a level that was closely related to the task concept. For each task, the robot needs a predefined set of skills. Each skill defines what the robot does on the system level. In other words, skills break the interaction down in functionalities that the robot can handle.

The advantage of using the concepts of skills and tasks together is that it is clearly defined which skills are needed for which tasks. The findings from the task analysis on the interaction level can be straightforwardly translated into changes that need to be made on the level of the skills. Thus, the skills should be modeled on a level that abstracts from the robot's components and the architecture.

The design of the skills depends on the kind of tasks that are implemented. As has been mentioned above, social tasks are largely responsible for managing the participant's attention. Thus, in the technical implementation, such tasks need to be translated into sensors, e.g., a Person Sensor which is a set of software components that enables the robot to detect users in its vicinity. The implementation of the functional tasks strongly depends on the scenario. However, most functional tasks probably require the skills to include actuators. The problem-related tasks will depend on strategies to solve the problems. How exactly these concepts are defined and used in the framework is described in the following.

4 Bonsai Framework

In this section, we explain the concept of Bonsai. We compare it to other systems to show what additional functionality and advantages Bonsai offers.

4.1 Concept

The main conceptual idea of Bonsai is to decouple the design and decomposition of the system into components

and the following architectural issues from the specification of system skills. While the architecture is still based on reusable components, we introduce the concept of abstract sensors and actuators, skills, and strategies in order to cope with necessary changes to the robot behavior without needing to reconfigure the components or architecture.

Bonsai is a domain-specific library implemented in JAVA that builds up on the concept of sensors and actuators that allow the linking of perception to action. The Bonsai sensors and actuators provide appropriate interfaces for the definition of skills and abstract from functional and hardware components or middleware issues. The sensors and actuators are used as atomic building blocks and encapsulate decoupled services provided by the software components of the system. A Bonsai sensor can be a simple abstraction of a real hardware sensor, e.g., a laser sensor, or a complex sensor such as the Person Sensor that is based on different software components to track persons in the robot's vicinity. We explain this sensor in depth, as it is the basis for every interaction with a user.

Figure 1 depicts the general structure of the Person Sensor. Again, the goal of this sensor is to provide information about a person in the robot's vicinity [17]. The readLastSeenPerson() method detaches the skill code (e.g., "follow person" skill) from the specific system configuration. In this case, the person sensor is internally realized by the configuration of four components that manage the detection and tracking of person hypotheses over time. This example addresses a Bonsai sensor. We will now also describe an example for an actuator. The Navigation actuator is a typical representative of a complex encapsulated actuator. As depicted in Fig. 2, the Navigation Actuator provides basic functions for a robot to navigate. These functions are expressed in different methods, e.g., the setGoal() method. Despite its simple interface, the Bonsai sensor triggers a rather complex interaction between software components. The interplay of the components dealing with the navigation, namely the SLAM component, the goal generator, the obstacle avoidance, and the path planner is transparent to the navigation actuator.

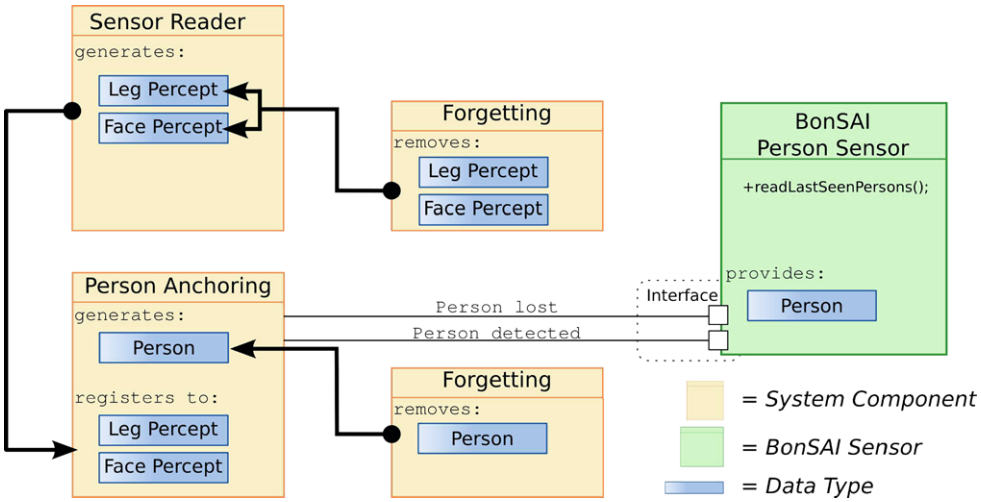

Fig. 1 The person anchoring sub-system with the system components on the left and the Bonsai Person Sensor on the right

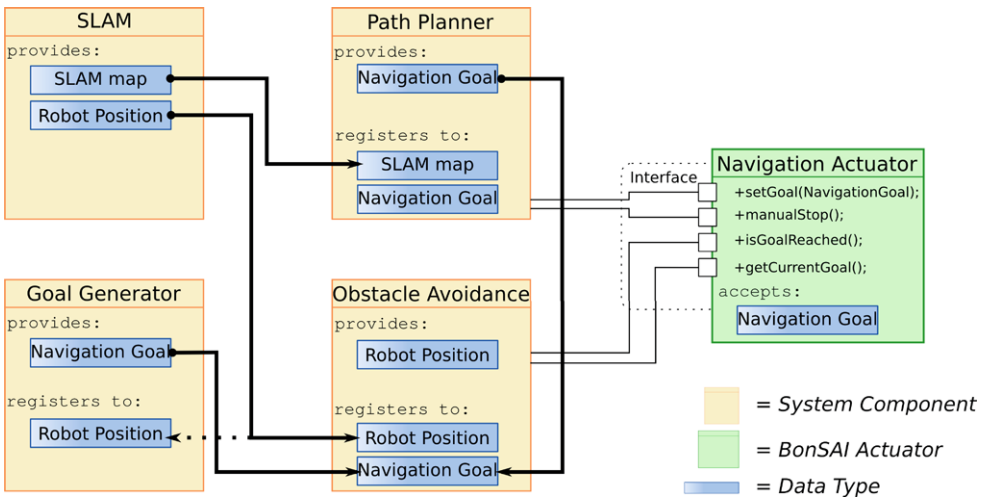

Fig. 2 The navigation sub-system with system components on the left and the Bonsai Navigation Actuator on the right

When calling the setGoal() method, the robot will move to a coordinate on the SLAM map that is declared in the NavigationGoal data while taking the obstacle avoidance into consideration. Thus, the actuator interface again detaches the skill code (e.g., "search for persons" skill) from the system's software components and the middleware by implementing a factory design pattern for sensors and actuators [11].

As the example shows, the sensors and actuators reach beyond simple hardware abstraction by encapsulating complex perception-action-linking processes. The ability of Bonsai to configure the system allows to have simple interfaces, e.g., the Person Sensor, which trigger a complex sequence of actions in the Functional Component level (see Sect. 5). One of the benefits of this approach is that a system configuration is linked to a specific skill of the robot. Skills are sequences of state-based deployments of sensors and actuators. As has been mentioned above, the sequence of activated sets of skills in the system corresponds to the task structure on the interaction level. The skills are combined to construct complex robot behaviors, e.g., the robot needs a greeting skill to initiate an interaction, throughout the interaction the person needs to be perceived, and the robot needs additional skills to complete the actual task such as learning about its environment. A skill can have multiple states that incorporate different sensors and actuators. A "follow person" skill for example starts with sensing people around the robot which it can follow. Then the robot announces this to the person and activates an appropriate strategy that sets the navigation goals related to the person. If the robot cannot perceive a person anymore, this is also announced and another strategy is used for the "navigation actuator". States like these (moving towards the navigation goal, moving in a way that allows to redetect the person, etc.) determine the robot's behavior. They are active until the necessary information for the next state is available, e.g., a person is detected.

In our current implementation, the states are controlled by an SCXML based state machine ([2], Apache Commons project: http://commons.apache.org/scxml) that builds up on the state chart formalism of Harel [14]. The state machine approach is very general and offers a standardized exchange format via SCXML. Bonsai itself does not model the high level control flow of a system and, thus, supports the decoupling of the skills from the control flow which is highly desirable because it ensures the reusability of the skills in different situations and scenarios. However, Bonsai could be used with other control abstractions, e.g., SMACH, which also works based on a state machine but is closely coupled to the ROS environment. However, SMACH does not explicitly model the behavior of a robot but rather is used for rapid prototyping of a scenario. For a more detailed comparison of the systems see Sect. 4.2.

The second important concept in Bonsai is the *skill* concept. The strength of the skill concept is that all similar situations during an interaction (e.g., all situations where a greeting is required) can be modeled and handled within one skill, even if necessary functionality is spread over different system components. We refer to this attribute of skills as being *local*. All information and possible actions are available from a skill (e.g., the greeting skill would have the information if a person was perceived through the sensors and provide different ways of saying the greeting). Thus, skills can also serve as observable units for evaluation on the interaction level. This behavior-oriented design (BOD) as, e.g., proposed by Bryson [5] enables the developer to model the behavior and the interaction instead of the system configuration.

The third important concept of Bonsai are *strategies* that bundle multiple Bonsai sensors in order to detect and solve problem-related tasks. Strategies only make use of sensors and produce output for one actuator (see Fig. 3). Following the concept of sensors and actuators that has been discussed above, the actuators are not limited to real hardware

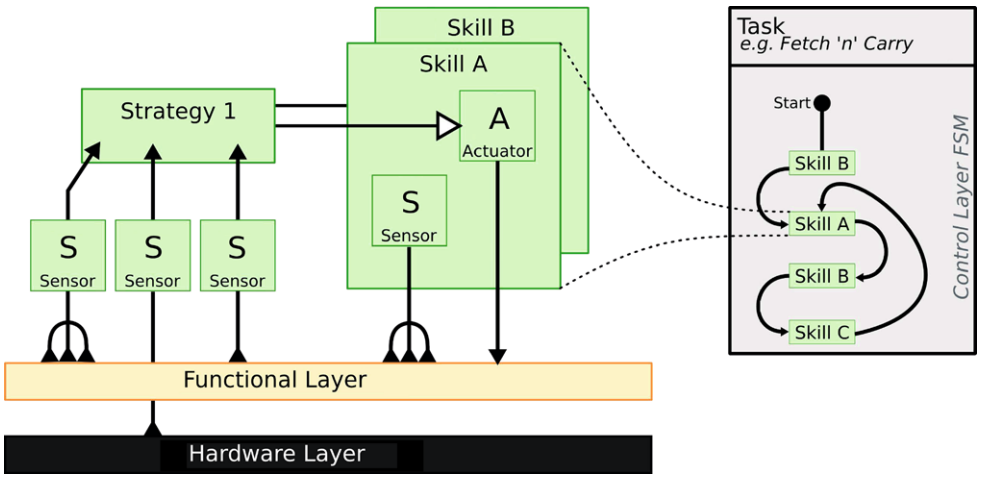

Fig. 3 Interplay between skills/strategies/sensors/actuators and the components (Functional Layer) and hardware (Hardware Layer) coupled with the schematic control abstraction of Bonsai

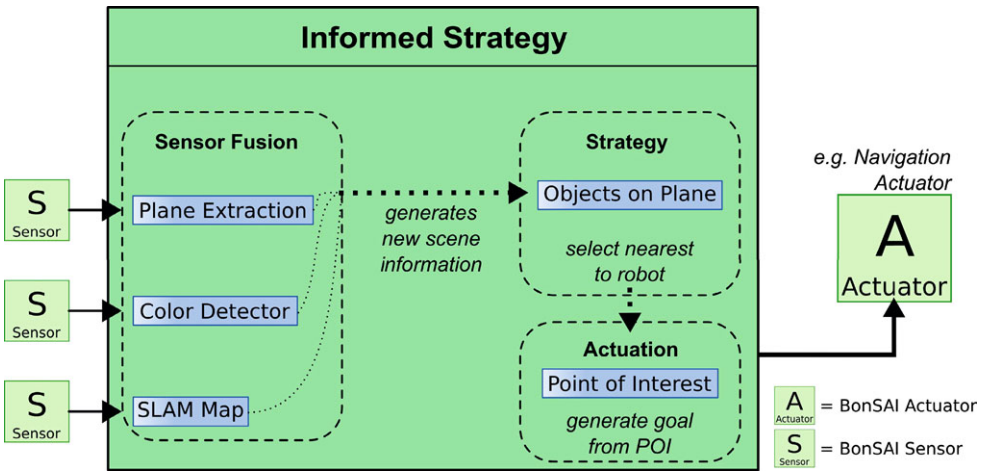

Fig. 4 Example for a Bonsai strategy that fuses the given sensor data to generate new information about the current scene and the objects in it, that are then used to select the nearest object to the robot (simple strategy) to generate a goal the robot can move to

actuators. Within the Bonsai framework, strategies enhance the re-usability of skills and accomplish an abstraction of how to process certain information available to the system. Strategies can be reused at different points in one skill or in different skills to react to unexpected situations during the processing. Strategies can trigger software components or sub-systems depending on the current situation. They therefore determine the way an actuator is controlled in Bonsai. Strategies can be "simple", e.g., following a person, where the distance the robot keeps between guide and itself is computed through a strategy. For more complex scenarios, strategies can also be a way to enrich the interpretation of the current scene. In that case, we speak of informed strategies. These perform a processing step on the data provided by the sensors (Sensor Fusion) which generates additional information that is used to guide the robot's behavior (see Fig. 4).

Assuming that one of the Bonsai sensors does not provide correct data, the strategy may detect an error and the system can react to that, e.g., by trying a different strategy. With the according code spread over several software components, the processing would fail with no chance to react. Thus, the Bonsai strategies support the loose coupling of the components that is desirable in interactive systems.

4.2 Comparing Bonsai with SMACH

As has been mentioned in Sect. 4.1, also other systems like SMACH enable developers to model robot behavior. SMACH is a good example of how prototyping environments for robotic systems often come together: An existing middleware abstraction is extended by a state-machine-based tool that allows to sequence the different software components to produce a consistent system action. This often does not involve a modeling of the robot behavior itself, which is also true for SMACH. It rather implements a stateful sequencer of the system components, which is sufficient to test features of the system components but is insufficient for modeling the robot behavior and to test, improve, or extend it over time. In fact, the typical way of rapid prototyping for robot scenarios as it is done with SMACH is counter productive for iterative system design since it focuses on system features and does not involve a proper behavior abstraction. With SMACH, the robot action and the control flow are modeled in the same place, the SMACH states. This reduces the reusability of the states and requires the developers to implement robot behavior either inside the controller or in the system components themselves. The strong dependency on the ROS middleware and the ROS messages intensifies this effect. Furthermore, for user-driven system design for interactive mobile robots the user and the interaction must be taken into account. This implies to have a continuous evaluation cycle with real users which highlights the importance of reusable skills and independence of the middleware or scenario. This is why Bonsai emphasizes the behavior abstraction and provides tools to model the *robot behavior* and the control flow separately.

4.3 Advantages of Bonsai

Bonsai has several advantages that we want to point out explicitly. The framework makes modeling new behaviors and adapting skills easy because the paradigm for our iterative design approach is to model the robot behavior locally. As mentioned above, this means that the strategies are part of the skills and are not spread over many components or rules. Thus, they are independent of the way the system architecture is implemented and if problems are identified in user studies, there is only one place where they need to be fixed—the Bonsai skill. Moreover, Bonsai skills can easily be transferred to other platforms and architectures. A first proof of this has been given in a bachelor thesis where one student implemented the framework in the NAO system within a short period of some weeks [34].

The behaviors implemented by skills are also minimal, e.g., only one functionality of the robot is modeled at the same time. Furthermore, they are modular to enable combination of many skills for a specific scenario. These three criteria (local, minimal, and modular behavior modeling) ease the iterative design process for developers, because they do not have to design rules for the component configuration, which requires a detailed knowledge of the system. Additional insights gained in user studies can be implemented into individual robot behaviors. These behaviors are then easily reusable for different scenarios or platforms.

5 Implementation of Bonsai in the BIRON System

In the following case study we apply Bonsai to improve a service robot in the home tour scenario. This scenario focuses on multi-modal HRI to enable the robot to learn about a domestic environment and its artifacts, the appearance and location of objects, and their spatial and temporal relations. In other words, the robot is guided through an apartment by the user and learns about rooms and objects. This is necessary because it is not possible to pre-program the robot for every potential environment. Hence, it has to learn about its surroundings with the help of the user. Once the robot has acquired the knowledge, it can serve as a kind of "butler" providing personal services (e.g., laying the table, cleaning rooms). In the current implementation, the learning is the prominent part of the scenario. However, new functionalities are being developed, e.g., the robot takes the user back to one of the objects or rooms learned previously. Thus, it could also show places to other people.

The possibility to incrementally extend the scenario by new tasks that require new robot skills is one of the advantages of the home tour. Many tasks are also transferrable to other domestic or public scenarios that require social interaction with single persons or groups of people. Thus, the scenario can be used to demonstrate the reusability of the skills. An example of this is the RoboCup@Home competition [42], where a collection of tests is defined and individually scored. Each test requires the robots to show a different collection of skills that are embedded in a human-robot interaction. As the tests are iteratively adapted or changed over the years, the re-use and extendibility of robot skills plays also a role. Last but not least, the scenario serves well to motivate repeated / long-term interactions which are a necessary requirement for an iterative design process.

5.1 Robot Platform

The hardware platform we use in the home tour scenario is BIRON, the BIelefeld Robot companiON. It is based on the research platform GuiaBot^{TM}, customized and equipped with sensors for analysis of the current situation in HRI. It runs autonomously and in soft real-time on two piggyback laptops. The robot base is a PatrolBot^{TM}. Inside the base there is a 180° laser range finder with a scanning height of 30 cm above the floor (SICK LMS) which we use for person detection, navigation, and map building. The cameras that are used for person and object detection/ recognition are 2MP CCD firewire cameras (Point Grey Grashopper). One is facing down for object detection and recognition, the second camera is facing up for face detection and recognition. BIRON is also equipped with an optical imaging system for real time 3D image data acquisition (SwissRanger) which is used for classification of rooms and objects. Additionally the robot possesses a Katana IPR 5 degrees-of-freedom (DOF) arm, a small and lightweight manipulator driven by 6 DC-Motors with integrated digital position encoders. The end effector is a sensor-gripper with distance and touch sensors (6 inside, 4 outside) allowing to grasp and manipulate objects weighing up to 400 grams. The upper part of the robot's body houses a touch screen (15 inch) as well as the system speaker. The on-board microphone has a hyper-cardioid polar pattern and is mounted on top of the robot. Finally, the platform is equipped with bumpers on both sides that enable a physical interaction. When pressed they will stop the robot immediately. The overall height of BIRON is approximately 140 cm.

5.2 Implementation

Brugali and Shakhimardanov [4] have pointed out that the configuration of the system, the connection between components at runtime, is crucial for component-based systems such as the robot BIRON. The configuration to a great extent defines what the robot is able to do at a certain point of time. This implies that the configuration needs to be dynamic to enable the system to react to changes in the environment and in the HRI task. A former architecture of the BIRON system covered this issue by introducing a component that could change the system configuration at runtime based on predefined rules [36]. While this approach made functionality of the robot feasible, it explicitly modeled component interaction instead of the desired robot behavior. With different scenarios and more complex tasks for a robot to interact in, the software architecture needs to focus more on designing the behavior of the robot and its interaction capabilities. This is why we implemented Bonsai in the BIRON platform.

The architecture of the BIRON system consists of many different components, each of which is a piece of software providing functionality, e.g., speech recognition, to the system. All components follow the concept of Information-Driven-Integration (IDI) [40] by sharing data via an active memory event-bus [41] within the system. Figure 5 shows

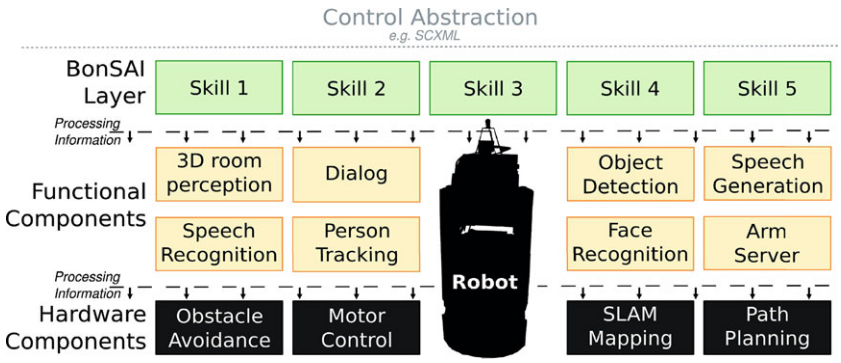

Fig. 5 Overview of the components of the BIRON system

Table 1 Overview of all Bonsai sensors and actuators

Sensors	Actuators
Laser	Navigation
Camera	Camera
Speech	Speech
Odometry	Arm
Position	Screen
Map	
Speed	
Object	
Person	

the different components that are used in BIRON. The different colors from bottom to top refer to the level of abstraction from the hardware.

Components in the dark gray level at the bottom either depend on direct sensory input or have a direct connection to the hardware. The obstacle avoidance for instance needs to get the input of the laser sensor at high frequencies to be able to detect obstacles while the robot is moving. The motor control represents a direct connection to the motors of the robot base to actually move the robot. Components on the yellow level (functional components) in the middle depend less on the robot's hardware and can facilitate information and/or functions provided by components of the layer below or of the same layer. The person tracking, e.g., facilitates information from the lowest layer (laser) as well as information from the face recognition. Based on these components, the Bonsai skills that were needed for the home tour scenario were implemented (green Bonsai layer). Examples for these skills are following a person, learning an object, or greeting someone. Table 1 gives an overview of all sensors and actuators implemented in Bonsai that can be used on BIRON. As has been mentioned above, the control of the resulting skills is not modeled in Bonsai. In this specific case we used the XCF middleware [41] and an SCXML state machine for coordination. However, Bonsai just as well works with other control abstractions and middleware.

6 Usability Test of the Bonsai System

After implementing Bonsai on BIRON, we conducted a usability study to ensure that the system does not only offer new functionality but also is usable and the functionality is accessible for the users. To enable a developer to understand how to produce a certain robot behavior was the most important factor in terms of usability that we took into account here.

6.1 Procedure

The usability study followed a standardized procedure for all participants. First they had to fill out a questionnaire about demographic information, experience with software development in teams and programming languages, experience with the robot BIRON and with developing behaviors for robots in general. After completing the questionnaire, the participants received two pages of instructions for the usability test. These included a short overview of the Bonsai framework and its building blocks as well as the actual programming task: The robot should wait in front of the entrance door until it is open. After that the robot should drive to the kitchen and check whether there was a person there. If a person was detected, the robot should announce that and leave the apartment.

All participants were provided with the same programming environment with an empty SCXML file. The simulation for testing was already running on the same computer. All necessary skills were provided in Bonsai with the only exception being the navigation. For the navigation the only help provided was where to look for example code and how the annotation looked like. For that purpose there were two additional help sheets with additional information.

After successfully implementing the task and testing it, the participants filled out a second questionnaire about their experience with Bonsai. Overall, the assignment took between 60 and 90 minutes.

6.2 Sample

The participants were selected among students and staff at Bielefeld University that had good Java knowledge and were familiar with XML. 17 people (16 male, 1 female) with an average age of 27.8 years (standard deviation (sd) = 3.28, ranging from 24 to 32 years) took part. All of them had developed software in a team before, none of them had used Bonsai with SCXML.

6.3 Results

One of the most important findings for us was that all participants finished the task successfully. The participants also indicated that they felt they had solved the task well (mean (m) = 4.24; sd = 0.66) on a scale of 1 (not at all) to 5 (very much).[1]) They understood the concepts behind Bonsai very well (understanding of the concept of sensors and actuators m = 4.41 (sd = 0.71); understanding of the concept of skills m = 4.35 (sd = 0.70)).

One of our main goals was to make Bonsai efficient and easy to use. To determine whether we achieved this goal, we asked users "how laborious did you find the development of a robot behavior in Bonsai" (m = 1.53; sd = 0.72); whether Bonsai saved them time when programming the robot behavior (m = 4.75; sd = 0.58); whether Bonsai supported them in programming (m = 4.31; sd = 0.70), whether Bonsai helped them to avoid programming errors (m = 3.75; sd = 0.77); and whether Bonsai enabled them to use the functions of the robot in a reasonable manner (m = 4.69; sd = 0.48). All mean ratings show that we succeeded in building a system that makes developing behaviors for robots efficient and easy.

Another important challenge for the Bonsai system was to develop robot behaviors in a way that makes them reusable. Therefore we asked the participants whether Bonsai supported them in programming reusable code (m = 4.31; sd = 0.87). The result shows that the developers strongly believe that the behaviors that they programmed were reusable. They also felt that the Bonsai system as such can be used on other robots/systems (m = 4.13; sd = 0.72). This finding implies that the robot behaviors are actually reusable across systems. Finally, the users agreed that Bonsai can be used to program complex scenarios (m = 4.38; sd = 0.72) which is a very important criterion for a system that shall enable robots to solve complex tasks.

From all these findings we conclude that Bonsai is actually a useful and efficient tool for developers of robot behaviors and that we succeeded in addressing the challenges introduced in Sect. 2 from a developer's point of view. However, this usability study did not address our main goal—the integration of system development and iterative user studies. This issue will be discussed in the following section.

[1]Please note that this scale applies for all following items.

7 Case Study

The case study is based on two user studies. The first user study was conducted to do a first evaluation of the robot, to design basic skills that it needs in the home tour scenario, and to identify types of changes that we will typically need to make to the Bonsai skills. After this study, the Bonsai framework was integrated in the robot BIRON (which was used in both studies). Thereafter, the robot with the framework was evaluated in a second user study and changes that needed to be made in the skills were identified. The paper shows how these changes can easily be made with the new framework and introduces metrics to quantify the amount of work needed to implement them (see Sect. 7.3).

The case study focuses on objective data of the interaction. The tasks and the structure of the tasks are analyzed in order to identify typical problems that occur. These problems avoid that the tasks are completed. We are aware that also the users' subjective impression of a task or, in other words, the user experience is very important (e.g., [39]). However, we have found that subjective data (interviews, questionnaires) of different users in first contact situations (situations in which the users interact with the robot for the very first time) is not very reliable. In other words, the users' subjective evaluation of the system often seems independent of the course of the interaction. For the two studies presented here, a comparative analysis showed that the users' evaluations were similar and largely independent of how well the robot functioned. One reason for this might be the novelty effect. First scientific inquiries in the novelty effect have been undertaken in the context of media for learning [7]. Clark describes the novelty effect as the tendency for performance to initially improve when new technology is introduced. This improvement, however, is not an actual improvement in learning or achievement, but is due to increased interest in the new technology. Transferred to HRI this means that people meeting robots for the first time are excited about the experience of interacting with the systems. This excitement often leads to higher effort in the interaction but also to evaluations of the robots and their abilities that are much better than would be reasonable for the course of the actual interaction. Moreover, the subjective data that we collected does not tell us, how the robot skills can be improved. From knowing that the users did not like something, we cannot infer how it needs to be changed to work better. Therefore, objective data is more useful for us in this context to show how the Bonsai framework supports efficient design of robot behaviors.

However, this does not mean that subjective data in general is worthless. On the contrary, when the data is acquired

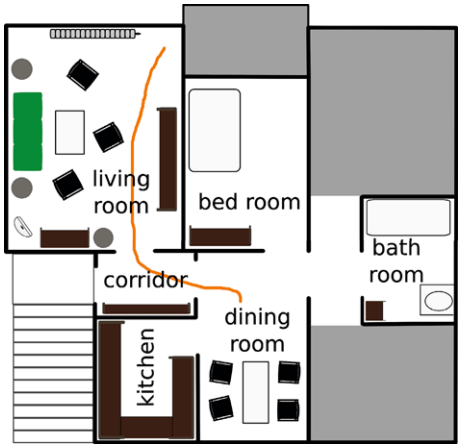

Fig. 6 Layout of the apartment where the user studies took place (*line* between living room and dining room = path the robot was guided)

from the same person in a long-term iterative design process it is extremely valuable and much more reliable.

7.1 User Study 1—Defining Requirements for Bonsai

The purpose of the first user study was to design skills in Bonsai and to identify typical changes that need to be made to the Bonsai skills. It was conducted on three days in November 2007. In the following, we describe the procedure of the study, the sample, the data analysis, and provide an example of the interaction. Thereafter, we present the results.

7.1.1 Procedure

To conduct the study in a realistic environment, people were invited to the robot apartment in Bielefeld (Germany) which provided an appropriate setting for the home tour scenario. The apartment is a typical German apartment (see Fig. 6).

The participants were welcomed and then received a written introduction to the study. The introduction consisted of a short description of the home tour scenario and the robot BIRON and an overview of the phases of the user study. Thereafter, the participants answered a questionnaire on demographic data and their experience interacting with robots. In user study 1, the users were then trained on using the speech recognition system, i.e., they were instructed about the proper placement of the headset microphone and were asked to speak some phrases for habituation. The recognition results were shown to them on a laptop. Thereafter, the participants were guided into the room where the robot was waiting ready for operation. The users were handed a tutorial script for practice to reduce hesitant behaviors. The script contained all commands they would need later on and the users were asked to try them out with the robot. This was necessary because the robot's vocabulary was restricted and we wanted the participants to have a feeling for the kinds of utterances that the autonomous robot understood. After the tutorial session, the participants carried out the main task. The instruction for this main task was:

- greet the robot
- guide the robot from the living room to the dining room via the hall (see Fig. 6)
- show and label the living room and the dining room
- show the bookshelf in the living room and the floor lamp in the dining room
- say good-bye

These tasks were chosen based on the scenario and covered all abilities that the robot had at this point of time. Taking into account that the robot was running fully autonomously, these tasks were quite complex. The participants were also free to try other things but these are not included in the analysis because they are not comparable between the interactions.

The whole interaction was videotaped with three cameras. One stationary camera was positioned in the kitchen and another one in the dining room. Moreover, we accompanied the interaction with a handheld camera. The experimenter who operated this camera was trained to not get in the way of the participants and always tried to keep an appropriate distance.

After the interaction, the participants were interviewed. They answered a questionnaire that included items on liking of the robot, attributions made towards the robot, and usability of the robot. More information about the procedure is provided in [24] and [25]).

7.1.2 Sample

This first study was conducted with 14 participants (9 male, 5 female). People with different ages and backgrounds were invited to the study. Their age ranged from 16 to 71 years (average 45.5 years). They were average computer users (3.2 on a scale of 1 [no experience at all] to 5 [a lot of experience]) and did have none or very few experience with robots (1.29 on the same scale of 1 (no experience) to 5 (a lot of experience)). All participants were native speakers of German and interacted with BIRON in German.

7.1.3 Data Analysis

As has been mentioned above, the whole interaction was videotaped. The videos were then imported into the annotation tool ELAN[2] where they were synchronized with logs from the interaction (e.g., speech understanding of the robot,

[2] http://www.lat-mpi.eu/tools/elan/.

person perception, utterances of the robot). Further annotations were added manually based on predefined coding schemes (e.g., speech of the user, tasks). The task annotation shall be discussed in a little more detail, as it is the basis for the analysis presented here.

The task annotation followed concrete rules. It was based on the verbal utterances of the user and the robot. For each task an opening utterance was predefined, e.g., the opening for an object teaching task was a user utterance like "This is a cup". Also the end of each task was predefined by utterances of the user or the robot. If the object teaching task followed the prototypical interaction script, it would end by the robot saying "Cup, I have taken a look at it". However, tasks might also be interrupted by other tasks. Either the user might ask the robot to do something else before the task is finished (e.g., to follow) or the robot might initiate a problem-related task (e.g., trying to detect the person again if it does not detect her anymore). Thus, only one task took place at a time and the interaction had a linear task structure as discussed in Sect. 3. As the annotation criteria were very well defined and did not leave room for interpretation it was not necessary to check for interrater reliability.

After being annotated, the tasks were evaluated with SALEM (Statistical AnaLysis of Elan files in Matlab), a toolbox that we developed for the purpose of analyzing ELAN annotations in Matlab [13]. The same toolbox was used for both studies, which ensured comparability of the data. SALEM is a quantitative approach to data analysis. We chose this approach to get a measure of how long and how many attempts the users needed to complete certain tasks. Moreover, SALEM provided us with a measure of how often problems occured in the different tasks and how long it took to recover from them. Thus, we could prioritize the tasks that needed to be adapted. SALEM was complemented by the qualitative SInA approach in order to find explanations for why problems in the tasks occurred and to identify possible solutions (see Sect. 3).

The main measures that we extracted were the number and mean duration (including standard deviation) of tasks per participant, the percentage of failures for each task, and transition probabilities between tasks. The results are presented later in this section.

7.1.4 Example Interaction

In the following we provide examples from the interaction to enable the reader to better understand the data analysis. As has been mentioned above, the progression of the interaction is determined by prototypical interaction scripts. To give an example, the prototypical interaction script for the greeting task in this user study consisted of the following steps [24] (il indicates that the step is taken on the interaction level, sl means that it takes place on the system level; see Sect. 3):

- user walks up to the robot (il)
- robot perceives user (sl)
- user receives an ID (sl)
- PTA[3] changes to state listen (sl)
- user says greeting (e.g., "hello", "hello Biron", "Biron hello") (il)
- robot processes utterance (sl)
- Mindi [animation on the screen] changes to thinking/processing (il)
- robot understands greeting (sl)
- PTA changes to state person (sl)
- Mindi changes to standing/lookingatuser (il)
- robot answers ("hello") (il)

Such a prototypical interaction script is specified for each task. From the example you can see that this particular script refers to specific components within the system (e.g., PTA, speech recognition, and the component that generates the display output). This implies that changes to the system would have to be made in all these components. In the Bonsai framework this is not necessary anymore (see Sect. 7.2).

The following example shows a typical interaction (the interaction level only which is determined by scripts as the one above). The utterances have been translated from German to English. If there is a line break between two utterances of the robot (R) or the user (U) this indicates that there was a long break or a non-verbal action between these utterances. Not all non-verbal actions are included in the example, as these would make it overly complex. If they are relevant to understand the example though, they are provided in brackets.

U: Hello Biron.
R: Hello. My name is Biron. What's your name?
U: Simon.
R: Nice to meet you Simon. Do you want to know something about me? Please say yes or no.
U: No.
R: Ok. You will see how everything works.
U: Biron follow me.
R: Ok, I follow you. (user guides the robot within the living room)
U: Biron stop.
R: Ok, I stop.
U: Biron, this is the living room.
R: This is the living room then. (Biron turns around to look at the living room)
R: Oh sorry, I can't see you anymore. Please say hello again if you still want to talk to me.
U: Hello Biron.
R: Hello.

[3]The PTA is the component for person tracking and attention (see [20]).

Table 2 Descriptive statistics of the task analysis of user study 1; count and percentage of failures that occurred in the tasks

Task	Count	Percentage count of all annotations	Minimum duration (seconds)	Maximum duration (seconds)	Mean duration (seconds)	Standard deviation duration	Duration of all annotations (%)	Count of failures	Percentage of failures
Social tasks									
greet	45	11.45	0.99	30.23	8.12	6.92	4.99	10	22.22
intro	5	1.27	19.09	57.50	41.81	18.08	2.86	2	40.00
farewell	18	4.58	2.41	23.15	9.13	5.61	2.24	4	22.22
Functional tasks									
guiding	109	27.74	3.28	75.07	22.84	12.75	34.00	9	8.26
teaching object	66	16.79	4.14	61.55	19.50	11.58	17.58	31	46.97
teaching room	40	10.18	3.87	37.36	17.71	9.04	9.68	10	25.00
Problem-related tasks									
register	38	9.67	8.19	55.45	19.19	11.55	9.96	4	10.53
obstacle	25	6.36	8.35	55.38	22.15	9.96	7.56	0	0
reset	18	4.58	13.26	92.14	28.15	19.39	6.92	5	27.78
stop	29	7.38	3.62	29.98	10.60	6.49	4.20	3	10.34
Overall	393	100	0.99	92.14	18.63	12.72	100	78	19.85

U: Biron, this is the shelf.
R: Ok, I'm taking a look at it.
R: Shelf, I have taken a look at it.
U: Biron follow me.
R: Ok, I follow you. (user guides the robot to the hallway)
R: Oh, I guess that there is some obstacle in my way. Please pull me away from it and say hello again. (user pulls the robot away from the obstacle (the door frame))
U: Hello.

This example is not complete but it includes all aspects that are relevant for the case study and the following analysis.

7.1.5 Results

The analysis is based on 393 tasks that were annotated for all participants in the study. Table 2 shows the descriptive statistics for these tasks and the number of failures that occurred in each task. All changes that needed to be made to the tasks are based on these quantitative findings and the qualitative analysis of interaction protocols like the one shown in the previous section. The changes were mainly motivated by the fact that almost 20 % of all tasks failed and another 28 % were problem-related tasks to repair the interaction (and even some of these failed).

We first describe the findings for all tasks. Table 2 shows that 45 *greeting* tasks were executed. The number is very high because in this study the participants obviously used the greeting to re-attract the robot's attention. The robot itself triggered the users to do so because part of the register task was that it asked the user to say hello again if she wanted to continue the interaction (see Example Interaction in Sect. 7.1.4). How this was handled will be shown in connection to the register task. One qualitative observation that we made concerning the greeting task was that BIRON's reply to the first greeting was "Hello. My name is Biron. What is your name?" The robot did not notice if the user had said "hello Biron". When the participants had used the robot's name in their greeting, it was quite obvious that they did not expect the robot to repeat it. The Bonsai skill should be designed accordingly.

After the initial greeting the robot offered an *introduction* to the users that most of them did not want to listen to. For the design of the Bonsai skill we decided that the users should not get the choice whether they wanted to listen to the introduction anymore. Instead the introduction should be shortened but everybody would have to listen to it. This approach was chosen to make sure that all users were reminded of the robot's abilities and to stress again that they can say stop at all times to make the robot stop whatever it is doing.

No changes needed to be made to the *farewell* task which basically consisted of saying good-bye. This task was the last one belonging to the social tasks. The tasks that are discussed next are functional tasks.

A main issue with the *guiding* task was that many users did not say stop at the end of the guiding which was a requirement of the prototypical interaction script. Therefore, the Bonsai skill was changed so that the robot now directly requests the user to say stop to finish the task and to enable the robot to switch to the next state. Also an additional Bonsai skill was designed to avoid this problem: the offer skill.

Table 3 Transition matrix for the most common functional (predecessor) and problem-related (successor) tasks in study 1 (*bold numbers* highlight the most frequent transitions)

Task	Predecessor		
	guide	teach object	teach room
Successor			
register	**0.211**	0.076	**0.200**
obstacle	**0.202**	0	0
reset	0.037	0.076	0.025
stop	0.018	**0.197**	0.125

If the stop command was recognized correctly and also all other tasks are finished, the robot should now ask what it should do next or how it could help. Thus, it should indicate that it is ready for the next command.

The quantitative analysis showed that the highest percentage of failures (47 %) occurred in the *object teaching* task. Two thirds of these were caused by speech recognition problems. The user labeled an object and the robot misunderstood the name. When the robot repeated the name at the end of the task (see Sect. 7.1.4), the user noticed the misunderstanding but could not repair it. Thus, we concluded that the Bonsai skill needed to include some verification step that allows the user to easily correct the robot. The same was true for the *teaching room* task. Additionally, the transition matrix (see Table 3) showed that this task in 20 % of the cases was followed by a register task. The main cause for this was that the robot turned around to take a look at the room which in most cases led to the problem that it could not perceive the user anymore and the user needed to register. Turning was necessary because the robot was not equipped with a 360° laser range finder but intended to acquire a representation of the whole room. It became obsolete with the improved component for navigation and mapping. This is positive especially because the participants did not understand why the robot turned and frequently tried to interrupt the turn.

Also the problem-related tasks were redesigned before they were implemented in Bonsai skills. The *register* task (making sure that the robot system perceives a user) was most frequent in this group (see Table 2). With respect to the development of Bonsai skills, we assumed that it would not be possible to design the robot in a way that makes the register task obsolete. Given that the users spent 10 % of the overall interaction time on this task, we decided that it would have to be more efficient though. The interaction example shows that the robot in the register task asked the user to say hello again if she still wanted to interact with it. We decided to avoid this in the Bonsai skill design and to equip the robot with the ability to resolve this situation on its own by adding a new Bonsai strategy. Thus, the robot should try to redetect the user by itself, only asking the user to wait to increase the chance of detecting her close to where she had been detected before. The analysis of the second user study will show how well the skill implemented based on this idea worked.

In the *obstacle* task, the user needed to pull the robot away from the obstacle because it could not free itself (see Example Interaction in Sect. 7.1.4). This usually happened when the robot tried to cross the door between the living room and the hallway that was quite narrow. Having users pull the robot is not acceptable especially not if the robot is used by elderly or physically impaired people. Therefore, the robot needs to be equipped with a behavior that enables it to solve these situations independently by turning and developing a strategy to get to its destination. This behavior was one actuator that was integrated in the Bonsai skill. Thus, the obstacle task was not necessary anymore.

The same is true for the *reset* task. This task allowed users to say 'reset' in order to reset the robot to a somewhat consistent state in order to being able to continue the interaction. All Bonsai skills should now be equipped with fallback behaviors in case of certain problems during the interaction. If none of the fallbacks takes effect, the system should be able to reset to a consistent state without a user command. Bonsai strategies have been implemented for the management of these failure states.

The last problem-related task was the *stop* task. The robot prompted the user to say stop because all tasks had to be completed. This restriction should be loosened in the Bonsai framework. The system should not initialize the stop task anymore because if the robot knows that it is waiting for a stop command, it should not have to ask the user for it. However, the users should still be able to use the stop command and to initialize the stop task themselves. This is a typical example of a Bonsai strategy that takes a speech sensor and triggers another software component.

7.1.6 Requirements for Bonsai

From the analysis we can summarize typical changes that we will need to make to the Bonsai skills based on user studies that will be conducted later in the design process. Changes to robot utterances will be necessary for various reasons: improving user experience (e.g., not confusing users by telling them the robot's name if they have just used it themselves), adapting choices that users are given (e.g., skipping tasks), or reminding users of things they have to say or do (e.g., asking them for the next command).

Moreover, the Bonsai framework has to support the implementation of new skills (e.g., the offer skill), of new steps within skills (e.g., verification steps that ensure that the robot has understood something correctly), and of changes to strategies (e.g., who has the initiative at a certain point of time). It is also necessary that skills can easily be removed if they are not necessary anymore (e.g., the obstacle skill that is obsolete because of more intelligent robot behavior). Based

on these findings, the Bonsai framework has been designed and tested in a second user study.

7.2 User Study 2—Using Bonsai on BIRON

When we conducted the second study on three days in March 2010, the Bonsai framework was already implemented. In the following, we analyze how well the Bonsai skills and strategies worked in the interaction and which additional changes needed to be made to them.

7.2.1 Procedure

The procedure of this second study was very similar to the first one. Again, it was designed to test all the skills of the robot that had by then been implemented. This includes the tasks mentioned above and two additional task: asking the robot to go back to one of the rooms or objects that had been learned previously. The high degree of similarity to the first study was necessary to allow for comparisons between both. In the procedure only one step changed: the introduction to the speech recognition system was skipped. This step was not necessary anymore because we were now able to use the on-board microphone of the robot for speech recognition. Apart from this, the procedure stayed the same (see Sect. 7.1.1). However, as has been described above, the prototypical interaction scripts had been changed based on the findings from the first study and the implementation in Bonsai. The effects of these changes on the interaction level are illustrated with an exemplary interaction in Sect. 7.2.4.

7.2.2 Sample

For the second study, we tried to keep the sample as similar as possible. We invited a new group of participants that had not interacted with the robot before. Again, 14 participants (7 male, 7 female) took part. Their ages ranged from 18 to 54 years (mean 38.9 years). Their experience with computers was very similar to what was indicated in the first study (3.1 on a scale of 1 [no experience at all] to 5 [a lot of experience]). The experience with robots (1.7 on the same scale of 1 (no experience at all) to 5 (a lot of experience)) was only slightly higher than in the first study due to one person who was more experienced. Again, all participants were native speakers of German and interacted with BIRON in German.

7.2.3 Data Analysis

The data analysis process was similar to the first user study (see Sect. 7.1.3). The videos were synchronized with manual annotations and logs from the interaction (e.g., speech understanding of the robot, person perception, utterances of the robot) in ELAN. One advantage that comes with Bonsai is that we can now log the skills and do not need to log components and later make sense of the different data.

The tasks were again annotated manually based on the same rules that were specified in the first study (see Sect. 7.1.3). They were also evaluated using the SALEM toolbox and the SInA approach. Given that each task consists of a predefined set of skills, future work should focus on automating the task annotations by extracting them from the skill logs.

The measures introduced above and in user study 1 focus on the interaction side of the system. However, we aim to also evaluate the improvements that came with Bonsai on the system side. It is difficult to measure system improvements with software metrics, because only a few, like the evolution matrix by Lanza [21], take the developing process and the whole system into account. The approach proposed by Lanza is a merely quantitative one looking at the evolution of classes by visualizing which classes grow/shrink, appear/disappear. However, this approach does not tell whether classes appear or grow because new functionality is embedded or because the system becomes inefficient and vice versa whether classes shrink/disappear because of software refactoring or because of removing functionality. Thus, the approach is not feasible here. Rather we have decided to use a simple measure based on Lines of Code (LoC) [9] that takes the lines of code that need to be modified for one change in a behavior and the number of different components involved into account. With these metrics we can show, how the effort of making changes in the iterative design process decreased with the new framework. An example analysis for the metrics is presented in Sect. 7.3.

7.2.4 Example Interaction

Also with Bonsai, prototypical interaction scripts determine the course of the interaction. They are implemented in the skills. We here show the new script for the greeting task for comparison:

- user walks up to the robot (il)
- Bonsai Person Sensor is initiated (sl)
- user says greeting (“hello”, “hello Biron”, “Biron hello”) (il)
- Bonsai greeting skill is initiated (sl)

The prototypical interaction script of the greeting is much shorter than it was before (four compared to eleven steps). However, it is of course still possible to make changes to the robot behavior and it is clear where these changes have to be made: in the Bonsai skill.

The following example shows how the new implementations of the tasks changed the interaction. As in the previous example, the utterances have been translated from German

to English. Again, if there is a line break between two utterances of the robot (R) or the user (U) this indicates that there was a long break or a non-verbal action between these utterances.

U: Hello Biron.
R: Hello. What's your name?
U: Simon.
R: Hello Simon. I can follow you and you can show me the apartment. You can say stop at all times, and I stop. What can I do for you now?
U: Follow me.
R: I will follow you until you say stop. (user guides the robot within the living room)
U: Stop.
R: I will stop here.
R: What shall I do?
U: This is the living room.
R: I will learn living room, right?
U: Right.
R: I have learned the living room.
R: What shall I do now?
U: This is a shelf.
R: I will remember the shelf, right?
U: Yes.
R: I have learned the shelf.
U: Follow me.
R: I will follow you until you say stop. (user guides the robot that is now able to go through the door autonomously)

This example illustrates the changes that have been made. They are now discussed in more detail.

7.2.5 Results

In the data of the second study, 385 interaction sequences were coded. Table 4 shows the descriptive statistics for all tasks and the number of failures that occurred in each task. The overall percentage of failures has decreased by 5 % (to 15 %) in comparison to the first user study. Even though this is not a lot on first sight and the difference is not statistically significant (one reason being that the standard deviation between the participants was very high with $sd = 10.33$ % in the first study and $sd = 8.61$ % in the second study), the result is positive given that two new functional tasks were introduced. We also compared the percentage of failures in the single tasks that were part of both studies. We conducted Fisher exact tests for failures and success for all tasks summarizing all participants. However, these did not reveal any statistical differences. Nevertheless, the findings can be backed up with qualitative observations which are of particular importance because the corpus is not very big.

Only very few failures occurred in the social tasks (greet, info, and offer) which indicates that the prototypical interaction scripts and the Bonsai implementations of these work well and no changes are required (see Table 4).

More failures occurred in the functional tasks. *Teaching objects* and *teaching rooms* are the most problematic tasks compared with the first user study. SInA revealed that in the teaching object task all failures were caused by speech understanding problems, in the teaching rooms task all but one.

Table 4 Descriptive statistics of the task analysis of user study 2, count and percentage of failures that occurred in the tasks

Task	Count	Percentage count of all annotations	Minimum duration (seconds)	Maximum duration (seconds)	Mean duration (seconds)	Standard deviation duration	Duration of all annotations (%)	Count of failures	Percentage of failures
Social tasks									
greet	14	3.64	8.51	103.26	28.43	24.45	7.22	0	0
intro	14	3.64	6.18	6.46	6.33	0.09	1.61	0	0
offer	133	34.55	1.06	19.50	3.00	2.82	7.24	4	3.01
Functional tasks									
guiding	69	17.92	0.96	103.16	26.78	24.58	33.54	12	17.39
teaching object	58	15.06	0.12	43.64	17.15	11.97	18.05	19	32.76
teaching room	47	12.21	0.36	61.18	16.20	10.49	13.82	14	29.79
showing object	8	2.08	8.10	169.32	60.76	53.11	8.82	2	25.00
showing room	6	1.56	5.08	86.96	29.84	31.59	3.25	5	83.33
Problem-related tasks									
register	20	5.19	2.16	59.34	12.29	13.93	4.46	0	0
stop	16	4.16	2.08	18.12	6.89	4.87	2.00	2	12.50
overall	385	100	0.12	169.32	14.31	19.34	100	58	15.06

These failures were very common because of the usage of the on-board microphone and point to the fact that speech recognition in natural environments is still a challenge. Nevertheless, not using a headset makes interaction in the home tour scenario much more natural and needs to be encouraged also in future studies. Therefore, as long as speech recognition technology does not become better, the problems have to be addressed by changing the robot behavior. Qualitative analysis revealed that the users are now (in contrast to the first study) able to tell the robot that it had misunderstood. However, this resulted in the robot saying that it would forget what it had just learned which counted as a failure even though the interaction could continue smoothly afterwards. As the robot should still learn the object or room, the more appropriate behavior would be to prompt the users to repeat the name. This change can easily be made in Bonsai by a local change in the corresponding state of the "learn new object/room" skill. Instead of treating the dialog component as a central control component, it is treated as a Bonsai actuator that provides an interface to the services offered by the dialog sub-system. Thus, a skill can request different dialog interaction patterns [31], e. g., changing it from an "information prompt" to an "information request". This example is further discussed in the context of the metrics for system redesign (see Sect. 7.3).

Also the *guiding* task resulted in failures. The percentage of failures for the guiding tasks actually increased from 8.26 % in the first user study to 17.39 % in the second user study. However, the analysis revealed that this was actually due to the fact that the guiding was less often interrupted by problem-related tasks in the second study (transition probability for guiding to the problem-related tasks 0.468 vs. 0.254). The users needed significantly less guiding tasks to reach the goal (mean number of guiding tasks 7.79 per user in user study 1 and 4.93 in user study 2; $p > 0.001$, Fisher exact test). Thus, the mean duration of the individual guiding sequences increased, increasing also the chance for the robot to fail, e.g., by misunderstanding a user utterance as a stop command or by the need to initiate a register task. To decrease the number of cases where re-registration of the user is necessary, the distance that the robot keeps from the user might have to be adapted. This change to the Bonsai skill is discussed as one example in Sect. 7.3.

Table 4 also depicts the results for the new functional tasks *showing object and room*. However, these will not be discussed here as they include only few cases and cannot be compared to user study 1.

The percentage of problem-related tasks overall decreased significantly from 28 % in the first study to 9 % in the second study ($p > 0.001$, Fisher exact test). Also the time needed to complete the problem-related tasks was much shorter in the second study. In the case of the *register* task, the mean duration decreased from 19.19 seconds to 12.29 seconds. The shorter mean duration of the register task is due to the fact that the robot had been enabled to find the user on its own without needing verbal input. The robot simply asked the user to wait for it, looked around, and signaled when it had detected the user by saying "Oh, there you are". In this case the participants probably did not take the disruption as seriously. However, this question has to remain open for future analysis. This new strategy worked with all users and all occurrences of the situation. The old strategy, that was also implemented when the person could not be re-detected, was never deployed. However, the difference in task length is not statistically significant which is again due to high standard deviations ($sd = 12.39$ seconds in the first study and $sd = 17.55$ seconds in the second study). In the first study most participants needed about 15 to 20 seconds to complete the task. In the second study one instance of the task took almost one minute whereas most others were completed within five seconds. If this one task is excluded from the test, it actually is highly significant ($T(47) = 2.7099$, $p = 0.0094^{**}$), indicating that the autonomous register task was actually quicker than the one based on verbal user input. Thus, we do not propose any changes to the Bonsai register skill at this point of time.

Table 5 Transition matrix for the most common functional (predecessor) and problem-related (successor) tasks in study 2 (*bold numbers* highlight the most frequent transitions)

Task	Predecessor				
	guide	teach object	teach room	show object	show room
Successor					
register	**0.254**	0.041	0.021	0	0
stop	0	**0.122**	**0.106**	0	0

The mean duration of the *stop* task was reduced significantly from 10.60 seconds to 6.89 seconds (for all stop tasks that did not fail $T(37) = 2.7716$, $p = 0.0087^{**}$). This shorter mean duration can be attributed to the changes of the prototypical interaction script of this task described above (see Sect. 7.1.5). Thus, the execution of this task seems to be quite efficient now and no redesign of the Bonsai skill is needed.

As in the first study, most problem-related tasks occurred during the guiding of the robot. Guiding sequences were followed by a register task with a probability of 0.254 (see Table 5). This happened slightly more often in the second study which means that person tracking while moving in space is still a problem, especially because lighting conditions in the apartment change between rooms. However, as stated before, the register task is solved more elegantly now because the robot had learned to re-register users on its own without asking them for help. Thus, there is no urgent need for changes in the Bonsai skill.

The teaching objects and rooms tasks were mainly interrupted by the stop task. SInA has shown that this was mostly due to the fact that the robot needed a long time to react to the users' utterances. Thus, the users initiated the stop task to make sure that the robot did not have a problem. As can be seen from the descriptive statistics (see Table 4), overall the stop task occurred less frequently in the second study (4 % of all tasks compared to 7 % in the first user study), however, the difference was not statistically significant. Given that the remaining stop tasks were initiated by the users only infrequently, no changes to the Bonsai skill are proposed.

7.3 Metrics for System Redesign

The results presented above focused on the interaction side of system design and changes that need to be made within certain skills have been identified. We also pointed out how making these changes profits from the Bonsai framework. In the following, we concretize the benefits of Bonsai on the system side by providing some examples based on the Lines of Code metric introduced in Sect. 7.2.3.

The first example concerns the guiding skill of the robot. As explained earlier, the robot should search for a person at the last known position before asking the user for help. Also the robot should announce right away if the person is moving too fast and the distance between person and robot increases too much ("Oh, please wait!"). This behavior is enabled via the information available to a Bonsai skill from other components and the code is not included in a component. Thus, the information is represented locally (see Sect. 4). In the prior BIRON system, e.g., the follow behavior code was included in one of the components, namely the Person Tracking and Attention (PTA) [20]. The control was achieved via a rule-based sequencer [36], which mainly had to switch between different components that would then control (parts of) the robot. With hardware abstracted in Bonsai via sensors and actuators we were able to decrease the complexity of the components since they do not need to include behavior code, hence being easier to debug/maintain, and we could increase the reusability of the developed behaviors via control abstraction, which allows to avoid control code, e.g., individually re-implemented finite state machines, in the Bonsai skills.

As has been mentioned above, the metrics used here to measure the effects of the implementation of the new framework are the lines of code that need to be modified for one change in a behavior and the number of different components where the changes need to be made. To make a simple change in the follow behavior in the prior robot system, e.g., the distance kept between person and robot, a developer would have to change two lines of code in the PTA component and four lines of code in the rule-based sequencer. These four lines concern two rules that control the robot behavior and one guard each that monitors the information exchange between components. Thus, overall six lines of code in two components have to be adapted. In Bonsai the developer changes one line of code in one skill.

A more complex example to illustrate the efficiency of the Bonsai approach is the re-inquiry of information, e.g., asking for the correct label of rooms and objects in case of a misunderstanding instead of forgetting the information and starting over. To implement this change without Bonsai, three components would have to be adapted: The dialog to initialize the user interaction, the rule-based sequencer, and the component that manages the information about the surrounding. The dialog would have to be extended by roughly three lines of code for the understanding part and the initialization of the user interaction. The sequencer would be changed in six lines: two new rules for the interaction with one guard each and two additional guards for the information flow between dialog and component. The specific component would be changed in three lines to achieve that the object label is not deleted but replaced. These changes of approximately twelve lines of code in three important parts of the system bare a high risk of undesired side effects. Additionally the behavior developer might not be an expert for the dialog or the component affected, which would make the changes even more difficult and risky. With Bonsai this change can be handled in one skill that needs five additional lines of code: Asking for the correct label (one line), replacing the wrong label (one line) and three lines for exception handling.

Going beyond these examples, it is easy to imagine that the number of lines that need to be changed in Bonsai can increase quickly for more complex changes in a behavior. However, also more lines in other components would need to be changed without Bonsai and the general rule for interactive systems is that it is better to change more lines in one place than to make a few changes in many places. This is due to the fact that changes in many components bare the risk of introducing undesired changes in the overall system performance and require a lot more knowledge about the overall system.

8 Conclusion & Future Work

The paper set out to design a framework that supports iterative system design for autonomous robots. From our point of view, such a framework needs to make adaptations of robot behavior based on findings from user studies efficient and needs to provide the possibility to re-use robot behaviors in future user studies and on different platforms.

We propose that these challenges can only be met with a framework that abstracts from the multitude of components within the system and allows to make changes in only

one place not requiring a deep knowledge about the whole system. Based on this assumption, we designed the Bonsai framework. The concepts for this framework were retracted from concepts that are also used in user studies because only in such studies we can determine concrete requirements for future system design. Furthermore, the paper shows the implementation of the framework in a robot. In a further user study we illustrate how the system works with the framework and how it supports iterative design. Thus, we contribute a system that supports iterative design by enabling researchers to adapt robot behaviors in an efficient way.

However, the efforts described here can just be the start of the iterative process of using Bonsai. In the future we aim to increase the usefulness of the framework by integrating more skills and strategies, by re-using skills in other scenarios and systems, and by using it in more iterations of user studies.

References

1. Adams JA (2005) Human-robot interaction design: understanding user needs and requirements. In: Proceedings of the 2005 human factors and ergonomics society 49th annual meeting, Orlando, FL
2. Barnett J, Akolkar R, Auburn R, Bodell M, Burnett D, Carter J, McGlashan S, Lager T, Helbing M, Hosn R et al (2007) State chart xml (scxml): state machine notation for control. In: W3C working draft
3. Bartneck C, Kulić D, Croft E, Zoghbi S (2009) Measurement instruments for the anthropomorphism, animacy, likeability, perceived intelligence, and perceived safety of robots. Int J Soc Robot 1(1):71–81
4. Brugali D, Shakhimardanov A (2010) Component-based robotic engineering. IEEE Robot Autom Mag 17(1):100–112
5. Bryson J (2010) The behavior-oriented design of modular agent intelligence. Agent technologies, infrastructures, tools, and applications for e-services. Lect Notes Comput Sci 2592:61–76
6. Burghart C, Holzapfel H, Haeussling R, Breuer S (2007) Coding interaction patterns between human and receptionist robot. In: Proceedings of humanoids 2007, Pittsburgh, PA, USA
7. Clark RE (1983) Reconsidering research in learning from media. Rev Educ Res 53(4):445–459
8. Clarkson E, Arkin RC (2006) Applying heuristic evaluation to human-robot interaction systems. In: Proceedings of HRI 2006
9. Fenton N, Pfleeger S (1991) Software metrics. Chapman & Hall, London
10. Fitzpatrick P, Metta G, Natale L (2008) Towards long-lived robot genes. Robot Auton Syst 56(1):29–45
11. Gamma E, Helm R, Johnson R, Vlissides J (1995) Design patterns. Addison-Wesley, Reading
12. Glas D, Satake S, Kanda T, Hagita N (2011) An interaction design framework for social robots. In: Proceedings of robotics: science and systems, Los Angeles, CA, USA
13. Hanheide M, Lohse M, Dierker A (2010) Salem—statistical analysis of elan files in matlab. In: Proceedings of workshop on multimodal corpora: advances in capturing, coding and analyzing multimodality, 7th international conference for language resources and evaluation (LREC 2010)
14. Harel D (1987) Statecharts: a visual formalism for complex systems. Sci Comput Program 8(3):231–274
15. Heerink M, Kröse B, Evers V, Wielinga B (2010) Assessing acceptance of assistive social agent technology by older adults: the almere model. Int J Soc Robot 2(4):361–375
16. Joosse M, Sardar A, Lohse M, Evers V (2013) Behave-ii: the revised set of measures to assess users' attitudinal and behavioral responses to a social robot. Int J Soc Robot 5(3):379–388
17. Jüngling K, Arens M, Hanheide M, Sagerer G (2008) Fusion of perceptual processes for real-time object tracking. In: Proceedings international conference on information fusion, Cologne, Germany
18. Kamide H, Takubo T, Ohara K, Mae Y, Arai T (2013) Impressions of humanoids: the development of a measure for evaluating a humanoid. Int J Soc Robot. doi:10.1007/s12369-013-0187-x
19. Kanda T, Shimada M, Koizumi S (2012) Children learning with a social robot. In: Proceedings of HRI, pp 351–358
20. Lang S, Kleinehagenbrock M, Hohenner S, Fritsch J, Fink GA, Sagerer G (2003) Providing the basis for human-robot-interaction: a multi-modal attention system for a mobile robot. In: Proc int conf on multimodal interfaces, Vancouver, Canada, pp 28–35
21. Lanza M (2001) The evolution matrix: recovering software evolution using software visualization techniques. In: Proceedings of the 4th international workshop on principles of software evolution, pp 37–42
22. Lee MK, Forlizzi J, Kiesler SB, Rybski PE, Antanitis J, Savetsila S (2012) Personalization in hri: a longitudinal field experiment. In: Proceedings of HRI, pp 319–326
23. Leite I, Martinho C, Paiva A (2013) Social robots for long-term interaction: a survey. Int J Soc Robot 5(2):291–308
24. Lohse M (2010) Investigating the influence of situations and expectations on user behavior—empirical analyses in human-robot interaction. PhD thesis, Faculty of Technology, Bielefeld University
25. Lohse M (2011) The role of expectations and situations in human-robot interaction. In: New frontiers in human-robot interaction. Benjamins, Amsterdam, pp 35–56
26. Lohse M, Hanheide M, Rohlfing K, Sagerer G (2009) Systemic interaction analysis (sina) in hri. In: Proceedings of conference on human-robot interaction (HRI)
27. Moratz R, Tenbrink T (2006) Spatial reference in linguistic human-robot interaction: iterative, empirically supported development of a model of projective relations. Spat Cogn Comput 6(1):63–107
28. Nalin M, Baroni I, Kruijff-Korbayová I, Cañamero L, Lewis M, Beck A, Cuayáhuitl H, Sanna A (2012) Children's adaptation in multi-session interaction with a humanoid robot. In: Proceedings of RO-MAN, pp 351–357
29. Nylander S, Ljungblad S, Villarreal JJ (2012) A complementing approach for identifying ethical issues in care robotics—grounding ethics in practical use. In: Proceedings of RO-MAN, pp 797–802
30. Payr S (2010) Closing and closure in human-companion interactions: analyzing video data from a field study. In: Proceedings of RO-MAN, pp 476–481
31. Peltason J, Wrede B (2010) Pamini: a framework for assembling mixed-initiative human-robot interaction from generic interaction patterns. In: SIGDIAL 2010 Conference, Tokyo, Japan
32. Quigley M, Gerkey B, Conley K, Faust J, Foote T, Leibs J, Berger E, Wheeler R, Ng A (2009) Ros: an open-source robot operating system. In: Open-source software workshop of the international conference on robotics and automation (ICRA)
33. Sarvadevabhatla RK, Ng-Thow-Hing V, Okita SY (2010) Extended duration human-robot interaction: tools and analysis. In: Proceedings of RO-MAN, pp 7–14
34. Schneider S (2009) Integration einer humanoiden Robotikplattform in einer serviceorientierten Architektur. Master's thesis, Faculty of Technology, Bielefeld University

35. Severinson-Eklundh K, Green A, Hüttenrauch H, Oestreicher L, Norman M (2004) Involving users in the design of a mobile office roboter. IEEE Trans Syst Man Cybern, Part C, Appl Rev 34(2):113–124
36. Spexard TP, Hanheide M (2009) System integration supporting evolutionary development and design. In: Human centered robotic systems. Springer, Berlin
37. Walters ML, Oskoei MA, Syrdal DS, Dautenhahn K (2011) A long-term human-robot proxemic study. In: Proceedings of RO-MAN, pp 137–142
38. Weiss A, Bernhaupt R, Lankes M, Tscheligi M (2009) The usus evaluation framework for human-robot interaction. In: AISB2009: Proceedings of the symposium on new frontiers in human-robot interaction, Edinburgh, Scottland, April 8–9, 2009, pp 158–165
39. Weiss A, Bernhaupt R, Tscheligi M (2011) The USUS evaluation framework for user-centered HRI. In: New frontiers in human-robot interaction. Benjamins, Amsterdam, pp 89–110
40. Wrede S (2008) An information-driven architecture for cognitive systems research. PhD thesis, Faculty of Technology, Bielefeld University
41. Wrede S, Hanheide M, Wachsmuth S, Sagerer G (2006) Integration and coordination in a cognitive vision system. In: Proceedings of international conference on computer vision systems, New York City, NY, USA
42. van der Zant T, Iocchi L (2011) Robocup@home: adaptive benchmarking of robot bodies and minds. In: Proceedings international conference on social robotics 2011. LNAI, vol 7072. Springer, Berlin, pp 214–225

Manja Lohse is a postdoctoral researcher in the Human Media Interaction group at University of Twente. Until June 2012 she was a member of the Research Institute for Cognition and Robotics (CoR-Lab) in the Hybrid Society Group at Bielefeld University where she received her Ph.D. degree in 2010. Manja Lohse's research interests are the social aspects of human-robot interaction, evaluation methods, and users' expectations toward robots.

Frederic Siepmann joined the CITEC Central Lab Facilities (CLF) in 2008 as a Ph.D. student. He is in charge of the research platform BIRON and a founding member of the RoboCup@HOME team ToBI. He participated in the German Open and the World Cup from 2009–2012 as one of the team leaders. His main research interests are in robot architectures and human-robot interaction.

Sven Wachsmuth is holding a Senior Lecturer position at Bielefeld University and is currently heading the Central Lab Facilities of the Center of Excellence Cognitive Interaction Technology (CITEC). He received the Ph.D. degree in computer science from Bielefeld University in 2001. In 2003, he spent a sabbatical year at the Computer Science Department of the University of Toronto. His research interests are in human-robot interaction, especially looking at high-level computer vision problems, and system integration and evaluation aspects.