FP6-IST-002020

COGNIRON

The Cognitive Robot Companion

Integrated Project

Information Society Technologies Priority

D7.2.1
Set-up of the Key-Experiment “Robot Home-Tour”

Due date of deliverable: 31/12/2004
Actual submission date: 31/01/2005

Start date of project: January 1st, 2004 **Duration:** 48 months

Organisation name of lead contractor for this deliverable:
UniBi

Revision: Final
Dissemination Level: PU

Executive Summary

In this report the current status of the key-experiment "Robot Home-Tour" is described based on the robotic platform and the overall setup as outlined in the "Key-Experiment Specification Document" (Deliverable D7.1.1). Within COGNIRON the "Robot Home-Tour" key-experiment (KE1) is intended to demonstrate those outcomes of the different research activities that relate to mobile platforms and multi-modal user interfaces. In the so-called *home-tour* scenario the basic idea is that a human introduces to a newly purchased robot all the objects and places in a private home relevant for later interaction. Here in this scenario, the multi-modal interaction situation is of fundamental importance and the users' attitudes and preferences can be evaluated with regard to the envisioned scenario of a robot as a companion for the home. Based on intermediate evaluations of the robot's performance in the home-tour scenario, the development of the individual COGNIRON Functions (CF) in the different research activities can be influenced. This feedback-loop between the research activities and the key-experiment will be of primary importance during the project to identify areas requiring further research and to improve the naturalness of the robot companion developed within COGNIRON.

1 Set-up of the Key-Experiment "Robot Home-Tour"

1.1 Development During the First Phase

In the first phase the basic setup of the key-experiment and the mobile platform used for demonstration was specified (see Deliverable D7.1.1). Based on the defined setup, an early stage of the key experiment allowing a limited dialogue interaction with the robot called BIRON was presented at the international workshop on Advances in Service Robotics (ASER'04, see [2] and the annex). Already in this first version it became clear that in order to coordinate a variety of different processing modules some kind of layered architecture with a central control component was necessary. To avoid concentrating all control in one place and achieve a modular system design, we realized a flexible control system that takes care of routing data between different components based on the overall system state. However, this so-called execution supervisor does not take decisions but only changes its status (and therefore the data flow between modules) based on data it receives from other modules. A comprehensive description of this component allowing for an evolutionary development of the robot companion's capabilities was presented at the international conference on Intelligent Robots and Systems (IROS'04, see [3] and the annex).
Throughout the first phase, discussions with the partners and the evolution of the different research activities have led to several iterations of the architectural definition w.r.t. the module interactions. The current definition of the overall component interaction is depicted in Fig. 1.
As the knowledge the robot acquires during interaction must be stored in a way that allows for efficient retrieval of this knowledge in later interaction situations, the current draft of the component interaction contains an abstraction in the form of a knowledge base manager (KB-manager). This abstraction will avoid that every individual component needs to have information about all different knowledge bases present in the system (for storing objects, locations, ...). Consequently, a component only needs to access the KB-manager that routes the request to the appropriate knowledge base. The implementation of this concept is underway and in the second phase first experiences with this abstraction will be available to possibly refine the way knowledge is being stored and retrieved in the overall robot companion.

1.2 Integration of Components from Individual Research Activities

For the implementation of the module interaction presented in the previous section, the components developed in the individual research activities have to be integrated in the key-experiment setup. With the central execution supervisor (see above) the component interaction can be realized, but this does require to cope with the technical challenges of coordinating different software components. For this purpose we apply the XML-enabled communication framework XCF (see [6]) that has been developed in the EU-funded project VAMPIRE for developing cognitive vision systems. The XCF framework is especially suited for the ongoing development process that will take place during the COGNIRON project as it allows for the flexible extension of data structures and supports distributed processing and active introspection. The combination of the architecture containing the central execution supervisor with XCF results in a flexible infrastructure for the development of robot companions that require an easy extensibility of their HRI-capabilities (publication accepted for ICRA'05, see [1] and the annex).

1.3 Current Capabilities of the "Robot Home-Tour" Companion

In the *Robot Home-Tour* scenario the human guides a newly purchased robot around in the house and introduces all objects and places to the robot that are relevant for later interaction. A first implementa-

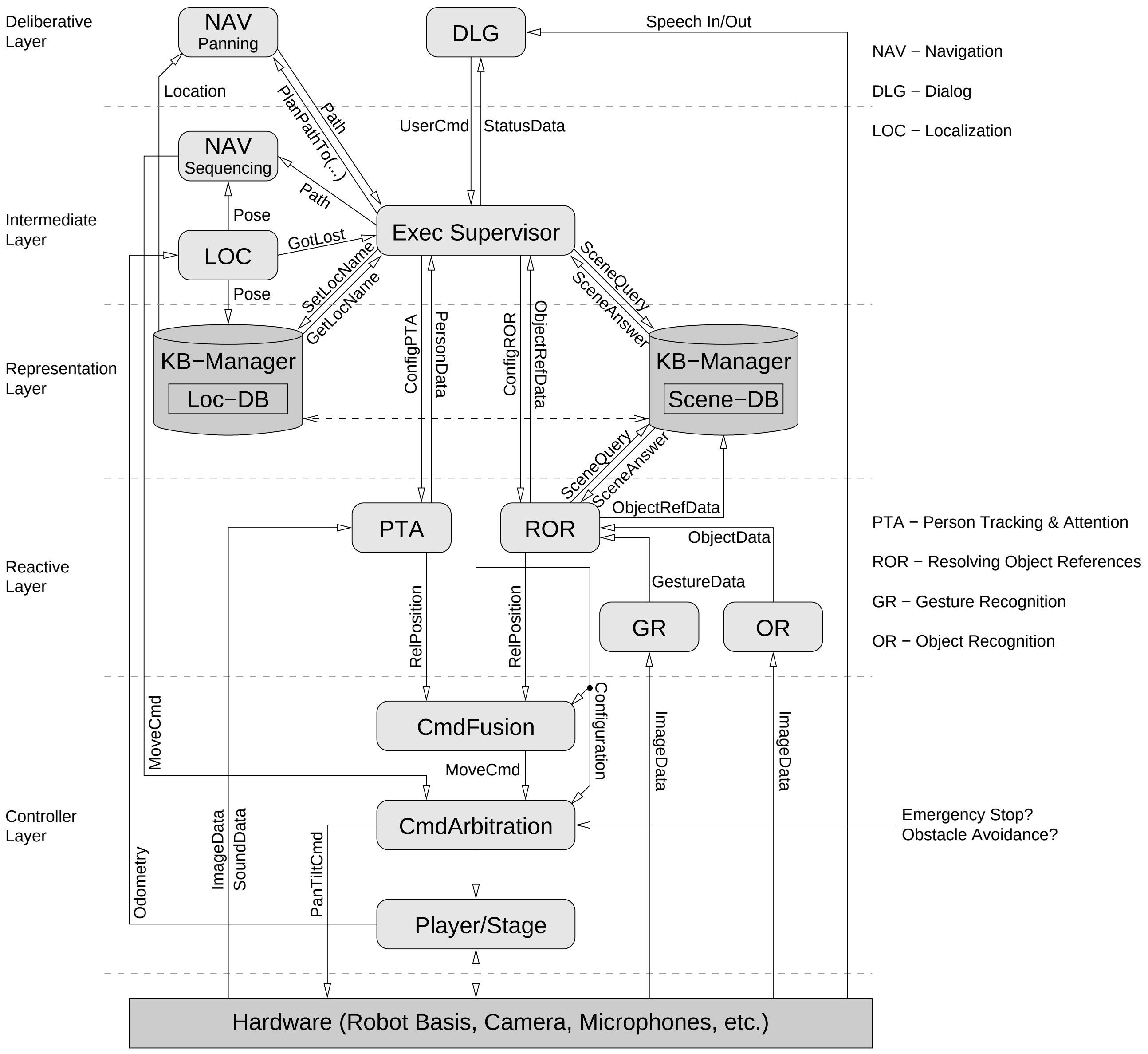


Figure 1: Module interaction for key-experiment "Robot Home-Tour"

tion of the home-tour key-experiment was realized based on the above outlined integration principles on the UniBi mobile robot platform BIRON. In this phase not all COGNIRON functions have been available for integration. Therefore, the functions actually used in the first demonstrator are intermediate versions of the functions that are being developed within the consortium. In the individual research activities improved versions of these functions are tested during the ongoing development in a conceptual integration by simulating the module environment. On the real robot, issues such as robustness to environmental changes and algorithmical stability are much more important. Although only a subset of the COGNIRON functions is integrated, the first demonstrator already gives a very good impression of the kind of interaction that will be possible.
At the current stage of integration, BIRON can track all persons standing or moving around the robot and focus its attention on the different humans. BIRON's capabilities allow a user to obtain the robot's

attention by looking at the robot and greeting it (e.g., by saying ”hello”). Subsequently, BIRON focuses its sensors on the person that has greeted the robot and accepts spoken commands. This verbal interaction can include a self-explanation given by BIRON of its current capabilities. Also, the user can ask the robot to follow him around through his home, which is an important aspect for the key-experiment. Objects in the environment can be shown to the robot by pointing. For example, the user can point to a plant and say ”This is a plant”. However, BIRON currently only processes the verbal information and does not have the COGNIRON functions for gesture recognition (CF-GR) and object recognition (CF-OR) as well as the overall component for resolving object references (CF-ROR) on-board that are required for multi-modal referencing of objects. First versions of these components will be incorporated in the next phase.
The current capabilities of the KE1 prototype have been demonstrated many times to visitors of the UniBi laboratory and first user studies have been performed in cooperation with RA1 (see [4]) to identify avenues for further research [5]. Besides the demonstrations within the lab, the KE1 prototype was presented at the EU IST Event in The Hague in November 2004. During the three days of the exhibition, many visitors interacted with BIRON and the current functionality of the home-tour demo platform was demonstrated to a large number of visitors including members of the European Commission.

1.4 COGNIRON Functions for KE1 “Robot Home-Tour”

1.4.1 CF-PTA: Person Tracking and Detection of Attention

This COGNIRON function allows the robot to always keep track of humans in its environment and to focus its attention on a specific person that became the communication partner by registering itself to the robot, e.g., by greeting it. Modalities used are laser range data (=legs), sound data (=speech) and vision data (=face, torso). The model of the human is rather coarse but allows to orient the robot’s base and camera in the correct direction to support gesture recognition (i.e., keep the human in the field of view of the camera). The person tracking and attention approach already shows a very good performance and is sufficiently robust to be used as basis for researching the interaction aspects. In the next phase a few more modalities will be added and especially identification aspects may be further improved.

1.4.2 CF-DLG: Multi-Modal Dialogue

In order to interact with the robot by natural speech, the current input analysis is capable of parsing simple and complete spoken command utterances. The currently integrated first version of the dialogue contains a simple sub-dialogue strategy with predefined sub-dialogues and provides speech-based system feedback. Right now this allows human-robot interaction with control of the robot sensors and the movement of the robotic platform. As the definition of the interfaces and the module interaction has been finished, an improved version of the dialogue can be easily integrated in the system by replacing the current component.

1.4.3 CF-LOC: Dialogue or Perception-Based Localisation

In the first phase the interfacing between the dialogue and the localisation component has been defined. This interfacing includes the integration of the localisation component with the central execution supervisor in order to allow other components to access the localisation information stored in a database. In the second phase the localisation component will be integrated on the robotic platform.

A wide angle camera will be used to obtain images of the environment. For this purpose the images from the stereo camera needed for gesture recognition can be used. The computing power required for localisation may be available by the on-board PCs depending on the necessary update rate. While the robot is moving, updates must be performed frequently but other components are likely to be idle (e.g., the dialogue). In case the computational power is not sufficient, an additional notebook will be mounted on the robotic platform.

1.4.4 CF-IA: Intentionality Attribution

The outcomes of this COGNIRON function is mainly informative and not in the form of a separate component that can be integrated. In the second phase results from user studies performed in the first phase about intentionality attribution will be available. These results will be incorporated in the development of the other COGNIRON components to provide the robot companion with the capability to express intentionality in terms of its multi-modal appearance.

1.4.5 CF-SOC: Socially Acceptable Interaction with Regard to Space

Similar to CF-IA, the outcomes of this COGNIRON function is mainly informative and not in the form of a separate component that can be integrated. The socially acceptable interaction with regard to space will influence several other components by providing, for example, information about adequate distances between robot and human during following. The person following behaviour that is currently running on BIRON is technically working fine, but it has not yet been adapted to also incorporate social aspects when calculating the robot's movements.

1.4.6 CF-ROR: Resolving Multi-Modal Object References

An additional multi-modal object resolving system using speech and visual input has turned out to be necessary to combine the results of gesture recognition (CF-GR) and object recognition (CF-OR). This functionality is closely linked to the dialogue but temporal aspects as well as interface issues require a separate solution. In the second phase a version of the system will be able to resolve references for known object types that are pointed at from a short distance (a few centimetres). For testing this function in the KE1 demonstrator, simplified versions of CF-GR and CF-OR will be used that can be implemented on the robotic platform in phase 2. These simplified versions will allow to test the dialogue interaction and the user behaviour when referencing objects in an early stage because the more complex functions that are being developed in the research activities cannot be used on a mobile platform, yet.

1.4.7 CF-GR: Gesture Recognition

The gesture recognition component relies on depth information from either a stereo camera or a depth sensor to recognise and classify gestures. In KE1 two types of gestures are of importance: command gestures and pointing gestures. As sensor the stereo camera mounted on the mobile platform can be used to obtain a rough depth estimate of the scene. However, real-time gesture recognition with the on-board PCs may not be feasible in the second phase, but is necessary to perform research on user behaviour when referencing objects in the environment of the robot (CF-ROR). Therefore, a simplified real-time gesture recognition to identify restricted 3D-pointing gestures will be used for CF-ROR. Command gestures may be recognisable in 2D and, consequently, such a component will be integrated in KE1 if its computational requirements can be satisfied on-board. Already in the first phase the necessary interfaces and module interactions have been defined to prepare this integration.

1.4.8 CF-OR: Object Recognition and Modelling

The object recognition function is a central component as it is needed by all key experiments. In this COGNIRON function a multi-modal (colour & depth) object detection including the estimation of the object's pose is being developed. However, in KE1 only a stereo camera and not a real depth sensor provides depth information of limited quality. Also, in KE1 the robotic platform does not allow the physical interaction with objects. Therefore, only a simplified version of CF-OR will be used in the second phase of KE1 to support research on the resolution of object references.

2 Future Work

While the key experiment ”Robot Home-Tour” is ahead of schedule as first functionalities can already be demonstrated, the definition of the data structures to be exchanged and the actual testing of the overall multi-modal functionality has turned out to be a major effort. While many problems have already been solved in the first phase, the ongoing integration of more components and testing of the resulting overall system is expected to require a substantial amount of person months. However, the integration of RA outcomes into the key experiment is closely linked to the software framework used for integration and requires a global knowledge of the other components in the scenario. Therefore, a single person with the majority of its person months dedicated to WP7.2 is considered very important for a successful demonstration of the key experiment home-tour.

3 References

3.1 Applicable documents

Deliverable D7.1.1: Specification of the COGNIRON Key-Experiments, 2004

3.2 Reference documents

[1] J. Fritsch, M. Kleinehagenbrock, A. Haasch, S. Wrede, and G. Sagerer. A Flexible Infrastructure for the Development of a Robot Companion with Extensible HRI-Capabilities. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Edmonton, CA, 2005. to appear.

[2] A. Haasch, S. Hohenner, S. Hüwel, M. Kleinehagenbrock, S. Lang, I. Toptsis, G. A. Fink, J. Fritsch, B. Wrede, and G. Sagerer. BIRON – The Bielefeld Robot Companion. In E. Prassler, G. Lawitzky, P. Fiorini, and M. Hägele, editors, *Proc. Int. Workshop on Advances in Service Robotics*, pages 27–32, Stuttgart, Germany, May 2004. Fraunhofer IRB Verlag.

[3] M. Kleinehagenbrock, J. Fritsch, and G. Sagerer. Supporting Advanced Interaction Capabilities on a Mobile Robot with a Flexible Control System. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, volume 3, pages 3649–3655, Sendai, Japan, September/October 2004.

[4] S. Li, M. Kleinehagenbrock, J. Fritsch, B. Wrede, and G. Sagerer. “BIRON, let me show you something”: Evaluating the Interaction with a Robot Companion. In W. Thissen, P. Wieringa, M. Pantic, and M. Ludema, editors, *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics,*

Special Session on Human-Robot Interaction, pages 2827–2834, The Hague, The Netherlands, October 2004. IEEE.

[5] B. Wrede, A. Haasch, N. Hofemann, S. Hohenner, S. Hüwel, M. Kleinehagenbrock, S. Lang, S. Li, I. Toptsis, G. A. Fink, J. Fritsch, and G. Sagerer. Research Issues for Designing Robot Companions: BIRON as a Case Study. In P. Drews, editor, *Proc. IEEE Conf. on Mechatronics & Robotics*, volume 4, pages 1491–1496, Aachen, Germany, September 2004. Eysoldt-Verlag, Aachen.

[6] S. Wrede, J. Fritsch, C. Bauckhage, and G. Sagerer. An XML Based Framework for Cognitive Vision Architectures. In *Proc. Int. Conf. on Pattern Recognition*, number 1, pages 757–760, 2004.

Annexes

The papers cited above that have been published as part of the work carried out in RA7 during the first phase of the project ([1]-[3]) are attached to this report.

A Flexible Infrastructure for the Development of a Robot Companion with Extensible HRI-Capabilities*

J. Fritsch, M. Kleinehagenbrock, A. Haasch, S. Wrede, and G. Sagerer
Applied Computer Science, Faculty of Technology
Bielefeld University
33594 Bielefeld, Germany
jannik@TechFak.Uni-Bielefeld.DE

Abstract— **The development of robot companions with natural human-robot interaction (HRI) capabilities is a challenging task as it requires incorporating various functionalities. Consequently, a flexible infrastructure for controlling module operation and data exchange between modules is proposed, taking into account insights from software system integration. This is achieved by combining a three-layer control architecture containing a flexible control component with a powerful communication framework. The use of XML throughout the whole infrastructure facilitates ongoing evolutionary development of the robot companion's capabilities.**

Index Terms— **robot architecture, human-robot interaction, system integration, distributed system, XML**

I. Introduction

In recent years an increasing number of mobile robots have been constructed. They are spanning a wide range in terms of their interaction capabilities starting from tour guides (e.g., [1], [2]) across service robots which are used in office environments and for fetch-and-carry tasks (e.g., [3], [4]) up to personal robots with a stronger focus on natural interaction capabilities (e.g., [5], [6]).

Nearly all of the realized personal robots are research prototypes that integrate a variety of individual software components but reach up to now only a limited interaction quality. However, for a commercial success personal robots need to become true companions with more sophisticated human-robot interaction (HRI) capabilities that make the interaction as natural as possible. In addition, such a robot must also be capable of adapting itself to unknown environments and, therefore, it has to be able to acquire new knowledge in a lifelong learning process. Moreover, as humans are around, reactive control of the robot's hardware is important, too. Our goal is to build a robot that satisfies all these requirements so that it can be accepted by humans in their private homes: a so called *robot companion* [7].

For the realization of any complex robotic system a modular approach is essential [8]. For a robot companion a large number of software modules implementing the different aspects relevant for a smooth interaction behavior have to be developed and integrated. Because this is usually an iterative process in large-scale systems, it must be possible to incorporate new modules and functionalities in a robot companion as they become available. Such a continuous extension of the robots functionality is usually very difficult as it not only requires modifying the control framework but also results in new data structures provided by the added components and possibly new control flows between modules. Therefore, the style of data flow in a modular system is also a crucial factor [9]. Another important aspect that is often neglected is the fact that the development of complex human-robot interaction architectures with many independent researchers involved is not only a matter of conceptual design but also a system engineering task.

We apply our mobile robot BIRON – the Bielefeld Robot Companion – to a specific application domain that currently gains increasing interest: the so-called *home tour scenario* (see also [7]). Here, a human introduces a newly bought robot all the objects and places in a private home relevant for later interaction between the human and the robot. For realizing such a system we have developed a generic robot control architecture to coordinate the activities of all integrated modules. The architecture allows a flexible extension of the overall system and is well suited to support natural human-robot interaction [10]. Our design decisions are motivated by earlier experiences gained from a previous implementation of the robot system [11] that lacked an advanced human-robot interface. The module communication on that robot had been realized using our former communication system DACS [12].

To provide a technical basis for building a robot companion, this paper focuses on the combination of our architectural methodology with the flexible, domain-independent, and easy-to-use XML-enabled Communication Framework (XCF) [13]. The resulting *System Infrastructure for Robot Companion Learning and Evolution* (SIRCLE) allows us to efficiently and transparently organize the realization and ongoing extension of our robot companion. We will show that through using XML as data format in the control component as well as for module communication the extension of the robot's software system with new modules can be realized very easily. This flexibility facilitates an agile software development process which has proven useful in software industry and results in a better ability to iteratively test intermediate versions of individual modules.

The paper is organized as follows: At first we discuss related work on robot architectures and their technical

*This work has been supported by the European Union within the 'Cognitive Robot Companion' (COGNIRON) project (FP6-IST-002020) and the 'VAMPIRE' project (IST-2001-34401) as well as by the German Research Foundation within the Graduate Programs 'Task Oriented Communication' and 'Strategies and Optimization of Behavior'.

realization in section II. Section III introduces our robot BIRON and in section IV we discuss requirements crucial for developing a robot companion. Subsequently, the XML-based communication framework is outlined in section V. Our robot control architecture and its implementation using XCF is described in section VI. Details of the control flow are presented in section VII and section VIII describes our experience with the proposed system infrastructure. The paper concludes with a short summary in section IX.

II. Related Work

There are several mobile robot systems that integrate capabilities for human-robot interaction in their architecture. For example, Care-O-bot II [5] is a multi-functional robot assistant for housekeeping and home care. Its modular control architecture uses a central execution module to which all other modules are connected. Low-level modules are implemented in C++, but the task execution module is implemented in Python and makes use of the network communication supplied within Python. *HERMES* [4] is a humanoid service robot and can perform fetch-and-carry tasks. Although its system's core is behavior-based, the robot has a situation-oriented deliberative component to pursue long-term goals. It consists of proprietary hardware and runs a self written operating system that allows sending and receiving messages via different channels [14], while overall control is realized by using a finite state machine. Jijo-2 [3] is intended to perform tasks in an office environment. Its system includes a reactive and an integrator layer, forming a hybrid architecture. The communication between the individual components is event-driven and based on low-level TCP/IP. Lino [6] serves as user interface to an intelligent home. To enable data exchange between all different components, a module-based software framework was developed, called the Dynamic Module Library. Here, each module has input and output ports that can be connected to each other to exchange data. On ROBITA [15], a robot that can participate in group conversations, all modules are connected by a priority-based coordination mechanism, which is based on a central blackboard [16]. A situation observation server monitors the blackboard and configures the overall system. This method allows to exchange and add new modules, but comes with the cost of the overhead for managing the blackboard.

Besides the architectures developed for the robots mentioned above there have been proposals for general robot architectures including application functionality. For example, BERRA [17], a three-layer architecture, is applied for robots performing fetch-and-carry and guiding tasks in the office domain. It is designed to provide scalability and a high degree of flexibility. Human-robot interaction covers mission acquirement only and, thus, user input is routed directly into a planner. The LAAS architecture [18] was originally designed for autonomous mobile robots, but is also used for other domains. It contains a central component, the so-called supervisor/executive, which coordinates data coming from the robot system and a planner, and commands from the operator. The focus is on the execution of valid and safe commands which are verified in hard real-time. The Tripodal schematic control architecture [19] is also three-layered and enables robots to fulfill transportation tasks or work as a tour guide [20]. The overall system has a central process supervisor and its configuration is modeled by Petri nets which is advantageous for realizing coordination of parallel processes. However, temporal synchronization is not explicitly modeled and is realized by an ad hoc solution.

The robot architectures mentioned above are different from pure robot control architectures primarily providing interfaces to a robot's hardware like, e.g., Player/Stage [21] or Sony's OPEN-R [22]. Most of the architectures presented in this section can be classified as hybrid [23], often using a centralized component to control the individual system modules. However, these architectures support only limited interaction capabilities that are far from a natural interaction. Moreover, their extensibility does not allow to incorporate a large number of HRI components that will be needed for reaching a high interaction quality.

III. Robot Hardware

The software infrastructure proposed in this paper is running on our mobile robot BIRON (see Fig. 1). Its hardware platform is a Pioneer PeopleBot from ActivMedia with an on-board PC (Pentium III, 850 MHz) for controlling the motors and the on-board sensors as well as for sound processing. A second PC (Pentium III, 500 MHz) inside the robot is used for image processing. An additional laptop (Pentium M, 1.4 GHz) is used for speech processing and dialog control.

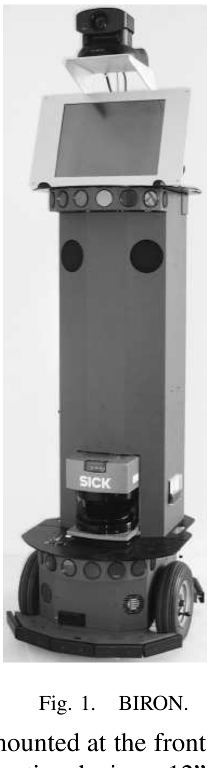


Fig. 1. BIRON.

The two on-board PCs are linked via an 100 Mbit Ethernet LAN switch that is also equipped with an 11 Mbit wireless LAN. The laptop is linked to this switch wirelessly, but can also be mounted on the robot for full autonomy. All three computers are running Linux.

A pan-tilt color camera (Sony EVI-D31) is mounted on top of the robot at a height of 141 cm for acquiring images of the upper body part of humans interacting with the robot. Two AKG far-field microphones which are usually used for hands free telephony are located at the front of the upper platform at a height of 106 cm, right below the touch screen display. The distance between the microphones is 28.1 cm. A SICK laser range finder is mounted at the front at a height of 30 cm. As additional interactive device a 12" touch screen display is provided on the robot.

IV. Requirements for Developing a Companion

To support progressive development of a robot companion's HRI capabilities, the functional requirements *mod-*

ularity, *communication*, *module coordination*, as well as *knowledge representation and acquisition* motivated in the introduction must be supported by the proposed system infrastructure. Additionally, several non-functional requirements play an important role that are discussed in the following.

One fundamental task for communication frameworks in the robotics domain is the ability to distribute modules across different processing nodes in order to guarantee fast system responses [24], [25]. This applies especially to large-scale systems like robot companions. However, most robotic researchers are no middleware experts, prohibiting the native use of, e.g., CORBA-based solutions. Communication frameworks built on top of CORBA [24] try to encapsulate complexity with domain-specific class libraries, which often complicates their use in system architectures of other domains. Thus, one requirement for a generic communication framework is the ability to enable researchers to easily build a distributed robot architecture. Additionally, it should allow for frequent integration cycles. To achieve these criteria, *simplicity*, *usability* and *standards compliance* are essential.

Furthermore, specifications change frequently in a research prototype. Thus an important feature is the *flexibility* of the communication framework. The impact of interface changes on an existing system architecture should be minimal to avoid the versioning problem [26]. Another important requirement that benefits directly from high usability and flexibility is *rapid prototyping*. Consequently, iterative development should not only be supported for single modules but also for the integrated system. Wrong directions in system evolution can more easily be identified if integration is performed on a regular basis starting at an early stage. For large-scale systems, software engineering research has shown that decoupling of modules is very important. Thus, the framework should support *low coupling* of modules. This facilitates not only independent operation of components but also minimal impact of local changes on the whole system. With a framework that adheres to low coupling, *debugging and evaluation* of a running system architecture can be supported more easily.

V. XML ENABLED COMMUNICATION FRAMEWORK

Taking into account the above mentioned non-functional as well as the basic functional requirements like module communication with flexible knowledge representations and data management, we developed XCF [27] in the context of cognitive vision and robotics. In the following we will present the fundamental concepts and show how these help to build an extensible robot architecture infrastructure.

Since it is very flexible, easy to learn and suited for abstract concept descriptions, XML was chosen as a basis to describe content transmitted, stored, and processed by the various robot modules. Note, that XML is not only used as a data exchange protocol as in XML-RPC-based solutions [25] that usually produce a lot of overhead through text-based representation of binary data and parameter encoding rules. Instead, we developed XML vocabularies for symbolic robotic data (e.g., states, events, objects, etc.). The instance documents then contain directly the semantical information that is accessed and selected using the standardized XQuery/XPath mechanisms.

Meta-information, e.g., about data types, is kept separate in the corresponding XML schema files and is not encoded in the instance document itself. Data type specification with XML schemas has several advantages in comparison to traditional programming language constructs. First of all, the data types are independent from specific programming languages. Even so, tools for using them are available for almost every platform. Furthermore, XML schemas are able to specify content models and ranges of allowed values in great detail. Providing fine grained sets of semantically grouped declarations in separate schemas with associated XML namespaces makes them reusable throughout different systems. Complex schemas for individual modules can then easily be composed out of these basic type libraries, only adding specific complex types. If taken into account, extensibility of data types is possible with schema evolution. Even complex grammars for components capable of interpreting and validating XML documents originating from different robot modules (e.g., the execution supervisor presented in section VII is an example for such a module) are easy to compose and well understandable with a sophisticated schema hierarchy.

Using XML for knowledge representation and schemas for data type specification, the Internet Communication Engine (ICE) [28] was chosen as technical basis of our framework. ICE offers similar functionality as CORBA, but with a much more lightweight approach. The ICE core manages communication tasks using an efficient protocol, provides a powerful thread mechanism and additional functionality that supports scalability. Similar to CORBA, ICE also relies on pre-compiled proxy objects. Unlike CORBA, ICE has no explicit dynamic invocation interface by itself.

On top of the ICE library XCF was developed to provide an easy to use middleware for building distributed object-oriented architectures that can efficiently exchange XML and referenced binary data (e.g., images). The referenced binary data structures are transmitted natively in a composite object together with the corresponding XML message. This combines the flexibility of XML with the efficiency of low-level communication semantics for large amounts of binary data. The XCF core itself features a pattern-based design and offers communication semantics like publisher/subscriber and (a)synchronous remote procedure calls/method invocations as well as event channels. All XCF objects and exposed methods can be dynamically registered at runtime.

Since data types are specified using XML schema as explained above, runtime type safety can be ensured. System introspection is directly supported through the implementation of the interceptor pattern [29] that helps in debugging and monitoring a running distributed system.

XCF conforms at least to the following transparency levels which are important for communication frame-

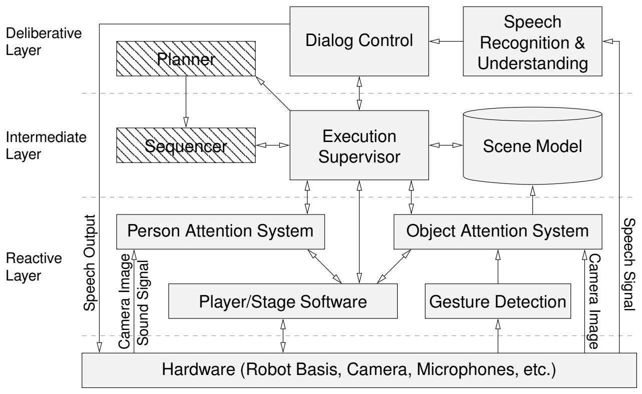


Fig. 2. Interaction architecture of BIRON.

works [30]: *Access transparency* is provided by the XCF core, where the implemented dispatcher service realizes *location transparency*. *Concurrent access* of multiple clients on one server is transparently handled by a specific worker thread pool. Additionally, the use of monitor threads provides *migration* and *error transparency* for computational modules.

To address the issue of data management, the active memory concept and implementation [31] is applied for use in our robotic framework. On top of a native XML database library [32], a server architecture was implemented that allows processing of the above mentioned data messages consisting of XML and referenced binary data. Thus, not only XML but also binary data can be shared by several robotic modules in parallel. For both kinds of data, powerful standard DBMS methods like insert, update, remove and query are exposed. Node selection and referral is based on XPath statements. Closely coupled to the knowledge representation a subscription model for distributed event listeners was realized, so that memory events can trigger registered processes that are interested in specific data and/or memory actions.

Though XCF is realized in C++ for performance reasons, there are also Matlab and Java bindings for rapid prototyping. It provides an easy to use basis for distributed processing in an asynchronous, decoupled fashion. Declarative, name-based selection of XML nodes with XPath expressions helps in building data types that can be extensible and in building systems that will not break if modifications occur. The following sections show how these foundations are used to realize a flexible and extensible architecture for human-robot interaction.

VI. Architecture

In principle our system is based on a three-layer architecture [33], as it is the most flexible way to organize a system which integrates autonomous control and human-robot interaction capabilities [10]. An overview of the resulting architecture can be seen in Fig. 2.

The most important component concerning the structure of the proposed architecture is a central *execution supervisor* (see section VII). The functionality of this execution supervisor is similar to the so-called *sequencer* used for ATLANTIS [34] where it coordinates the operations of the modules responsible for deliberative computations rather than vice versa. This is contrary to most hybrid architectures where a deliberator continuously generates plans and the reactive plan execution mechanism just has to make sure that a plan is executed until a new plan is received.

The main modules in the deliberative layer are the planner and the dialog control. The planner is responsible for generating plans for, e.g., navigational tasks. The dialog control carries out dialogs to obtain instructions given by a human interaction partner via the speech understanding system [35]. It is also able to manage interaction problems and to resolve ambiguities by consulting the user. The dialog control sends valid instructions to the execution supervisor which is located in the intermediate layer of our architecture. Because the dialog control is directly connected to the central component, ambiguities that might arise from modules in the reactive layer can also be resolved by dialog. For this purpose corresponding enquiries from the reactive layer are routed through the execution supervisor. Thus, results from HRI are made available centrally in the architecture instead of routing them only to a planner for pure mission acquirement like it is done in many other robot architectures (see, e.g., [17], [19]).

The sequencer also resides in the intermediate layer of our architecture. It is responsible for decomposing plans provided by the planner, as the execution supervisor can only handle single commands. A scene model for maintaining knowledge representations completes this layer.

The person attention system [36] represents the robots main reactive feedback control mechanism and is therefore located in the reactive layer. It detects potential communication partners among persons present in the vicinity of the robot. It is configured by the execution supervisor to show different behaviors, e.g., to look at all people in its surrounding or to track a specific communication partner. However, the person attention system does not directly control the robot's hardware. This is done by the *Player/Stage* software [21] which provides a clean and simple interface to the robot's sensors and actuators. Even though we currently use this software to control the hardware directly, the controller can easily be replaced by a more complex one which may be behavior-based to also include, e.g., obstacle avoidance.

Besides the person attention system an object attention system is located in the reactive layer. The execution supervisor can shift control of the robot from the person attention system to the object attention system in order to focus objects referred to by the user. The object attention is supported by a gesture detection module which recognizes deictic gestures [37]. Combining spoken instructions and a deictic gesture allows the object attention system to acquire visual information of a referenced object. This information is sent to the scene model in the intermediate layer.

The scene model stores information about objects introduced to the robot for later interactions. This information

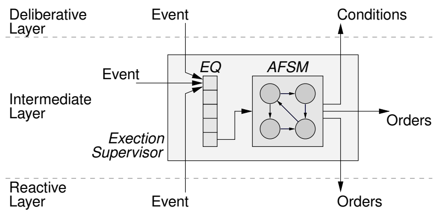


Fig. 3. Schema of the execution supervisor. It contains an event queue (EQ) and is controlled by an augmented finite state machine (AFSM).

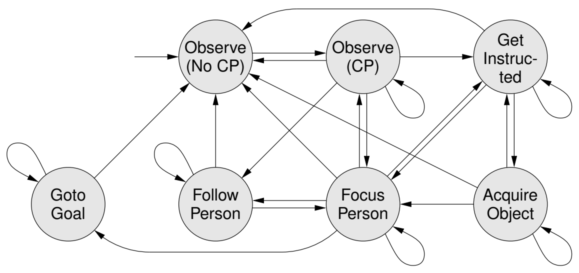


Fig. 4. Augmented finite state machine of the execution supervisor.

includes attributes like position, size, and visual information of objects provided by the object attention module. Besides, additional information given by the user is stored in the scene model, e.g., a phrase like "This is my coffee cup" indicates owner and use of an object to learn.

In order to satisfy certain system safety requirements, modules should fail perceivably as in a real world robot failures cannot be excluded. We realized this feature by messages which are initiated by the main loop of each module and sent in fixed intervals to modules being in communication with this module. If a module does not receive these messages anymore, it can determine that the corresponding sender stopped working correctly. In this case corrective actions can be taken to recover from the failure. If recovery is not possible then the robot is at least able to ask the user to call technical support.

VII. Execution Supervisor

The execution supervisor is the central part of our architecture and is designed to be as generic as possible. In order to achieve this requirement the execution supervisor interprets no data at all: It either configures modules of the system at runtime-based on received events containing needed parameters, or it routes specific data to modules which are responsible for processing the data. Since all information is carried by XML documents, the execution supervisor even does not need to distinguish between the data structures it receives.

The execution supervisor receives data in the form of events from all modules connected to it. Because these events occur asynchronously and latencies in the communication have to be considered, the execution supervisor uses an *event queue* (see Fig. 3). All incoming events contain timestamps indicating when they were created. They are inserted in the queue ordered by their timestamps. The events are handled in turn, starting with the oldest one.

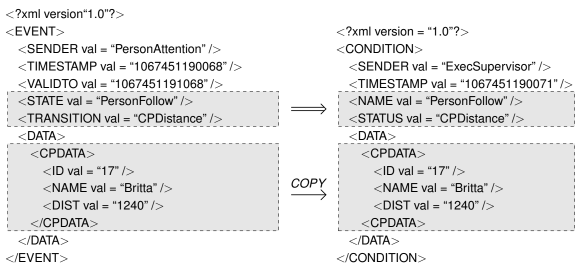
The event queue is also used to synchronize events. For example, a person can only become the robot's current communication partner if the person attention system signals that it has detected a potential communication partner and the dialog control notifies the execution supervisor that it has received a corresponding speech input. Only if both events arrive at the execution supervisor in a certain interval of time, it is assumed that these events belong together. This is verified by a lifetime entry in the corresponding events which describes how long an event remains valid. For a more detailed explanation please see [10].

In order to process the events stored in the event queue, the execution supervisor is controlled by an *augmented finite state machine* (AFSM). Each event corresponds to a transition in the AFSM. Thus, transitions from one state to another can only be triggered by events. When a transition is executed, the corresponding event is deleted from the event queue. As the execution supervisor is not intended to interpret any data, the AFSM is not very complex (see Fig. 4). Therefore, the structure of the AFSM is also specified in XML. As a consequence, no special-purpose language like, e.g., RAPs [38] is necessary. Defining the AFSM in XML results in a clear representation which allows to quickly restructure or extend the execution supervisor without recompiling it. This concept also allows us to modify the execution supervisor at runtime. As new XML documents can be handled directly by default their integration is straightforward.

The finite state machine is augmented in so far, that with every transition which is executed, a specific *action* is performed. These actions are for configuring the system by emitting two specific types of messages: *conditions* and *orders*. These messages include parameters needed by the addressed receivers. The parameters are supplied by the event which initiated the corresponding action. Orders are sent to all modules in the intermediate and reactive layer in order to reconfigure the system. Conditions are sent to modules in the deliberative layer which inform these modules about the internal state of the overall system. This form of communication reflects the hierarchical structure of the architecture: Orders are sent 'downwards', while conditions are sent 'upwards'.

An example on how the execution supervisor routes information from one module to another one can be seen in Fig. 5. Here, the execution supervisor receives an event from the person attention system, which contains new data about a communication partner. Depending on the overall system state, this event initiates a specific condition, which is then sent to the dialog control module. A copy of the data from the event is included in this condition.

Altogether, the execution supervisor processes events from different modules and also ensures that certain events have to arrive in a certain time interval before the contained AFSM changes to a specific state. This reflects that the execution supervisor is in control of the overall system. The dialog control agent can give advices, but if the components of the remaining layers do not provide corresponding



```
<?xml version"1.0"?>
<EVENT>
  <SENDER val = "PersonAttention" />
  <TIMESTAMP val = "1067451190068" />
  <VALIDTO val = "1067451191068" />
  <STATE val = "PersonFollow" />
  <TRANSITION val = "CPDistance" />
  <DATA>
    <CPDATA>
      <ID val = "17" />
      <NAME val = "Britta" />
      <DIST val = "1240" />
    </CPDATA>
  </DATA>
</EVENT>
```

```
<?xml version = "1.0"?>
<CONDITION>
  <SENDER val = "ExecSupervisor" />
  <TIMESTAMP val = "1067451190071" />
  <NAME val = "PersonFollow" />
  <STATUS val = "CPDistance" />
  <DATA>
    <CPDATA>
      <ID val = "17" />
      <NAME val = "Britta" />
      <DIST val = "1240" />
    </CPDATA>
  </DATA>
</CONDITION>
```

Fig. 5. Routing example: Data of an event from the person attention system are propagated to the dialog control via a specific condition.

information advices are rejected. For example, the robot may perceive instructions from a radio, but it will not start an interaction as no person can be detected. This makes the interaction with the overall system more robust.

Due to the low complexity, our concept of the execution supervisor also scales up if the system is extended. Generally, only one more state is needed when a new functionality or module is added.

VIII. Experience

The presented system infrastructure SIRCLE was successfully applied to our mobile robot BIRON. With several researchers contributing to this system, the proposed solution proved its suitability for the realization of the complex robot architecture presented in section VI.

The module integration task directly benefits from the fact that XCF is very easy to use and the applied concepts are highly standards-based. Furthermore, a central system view utilizing the active system introspection supports debugging and evaluation of the running system.

Loose coupling of modules and the declarative style of accessing system data paid off in ease of modification and the ability to let the system architecture evolve over time as we integrated new modules and data types. For example, a first prototype of the object attention system was added just recently and required on the architectural level only a modification of the execution supervisor's configuration file and thus demonstrating iterative development. Additionally, loose coupling of modules as well as error and migration transparency yield increased robustness of the overall system.

With respect to the performance of the overall system, the distribution and coordination of modules using SIRCLE has resulted in a low response time allowing for more natural human-robot interaction. In order to obtain quantitative results of the overall system behavior corresponding time measurements were carried out using a typical interaction example. The results of this evaluation are presented in Fig. 6 in a sequence diagram. All interactions with BIRON follow this pattern in principle, even though the duration for the speech processing might vary. The main reason for this is that the time needed to process speech input depends on the length of the utterance given by the user.

In Fig. 6 it can be seen how much time is needed for the most important processing phases and that the speech

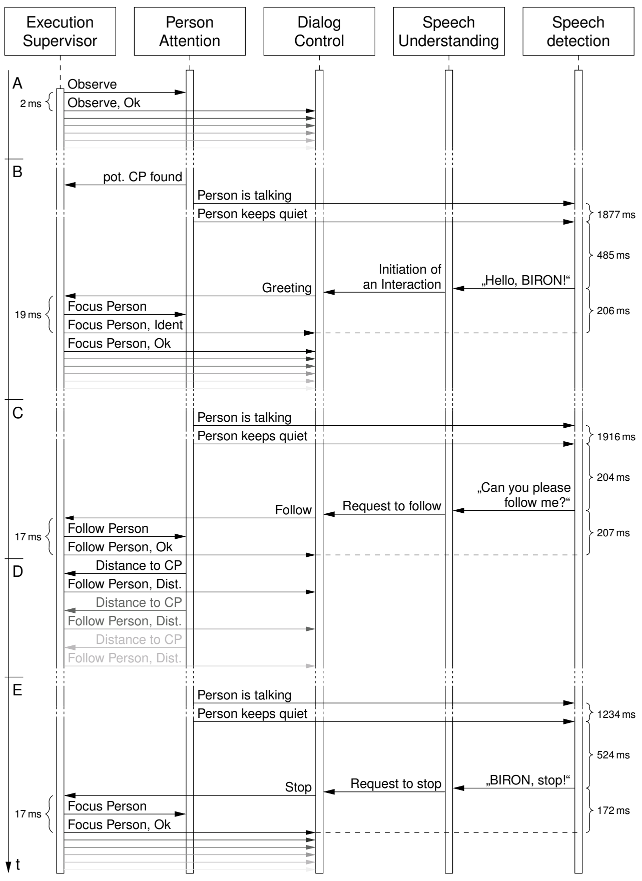


Fig. 6. Qualitative Representation of the time course of the module communication on BIRON considering an interaction example. Arrows getting brighter indicate messages that are sent continuously. Different phases: A: The system is started. B: A person approaches BIRON and greets the robot. C: The person asks BIRON to follow her. D: BIRON follows the person. E: The person asks BIRON to stop following.

detection takes most of the processing time. Subsequently the system usually needs only around 200 ms until it produces an answer to the user. According to this duration, the execution supervisor takes only a very small amount of time to reconfigure the system: less than 20 ms. Time needed for module communications is also very small and shows that the advantage of increased processing power in a distributed system exceeds the cost for module communication. Transferring a message from one to another module takes only a few milliseconds in average, but depends on whether it is routed via WLAN or not. A more fine grained examination is difficult as time measurements slow down the system and falsify results. However, the results show that the software infrastructure on BIRON enables efficient module interaction that leads to a reactive overall system.

The suitability of our system infrastructure SIRCLE was proven by presenting two BIRON robots at the exhibition of the Information Society Technologies Event (IST Event) in November 2004 in The Hague, The Netherlands [39]. The robots where presented to the public at all three exhibition days, twice for 9 hours and once for 6 hours. They where instructed to either follow the user or to look at some objects. The robots worked continuously and robust.

Altogether visitors watched the presentation of the BIRON systems with great interest and the performance of the robots was consistently denoted as impressing.

IX. Summary

In this paper we presented SIRCLE, a system infrastructure providing a software platform for a robot companion exhibiting powerful capabilities in human-robot interaction. Our approach combines the XML-enabled Communication Framework presented in section V with our three-layer architecture for controlling module operation and communication as shown in sections VI and VII. The overall concept proved its suitability during the ongoing work on our robot companion BIRON. Targeting not only the functional requirements, our infrastructure features simplicity, standards compliance, and extensibility which results in an agile development process and, ultimately, in a robot companion with more natural human-robot interaction capabilities. The suitability of our approach was proven by a quantitative system evaluation and the successful presentation of our two BIRON robots at the IST Event 2004.

References

[1] S. Thrun, *et al.*, *Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva*, *International Journal of Robotics Research*, vol. 19, no. 11, pp. 972–999, 2000.

[2] N. Tomatis, *et al.*, "Building a Fully Autonomous Tour Guide Robot: Where Academic Research Meets Industry," in *Proc. Int. Symp. on Robotics*, Stockholm, Sweden, 2002.

[3] H. Asoh, *et al.*, *Jijo-2: An Office Robot that Communicates and Learns*, *IEEE Intelligent Systems*, vol. 16, no. 5, pp. 46–55, 2001.

[4] R. Bischoff and V. Graefe, "Demonstrating the Humanoid Robot *HERMES* at an Exhibition: A Long-Term Dependability Test," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems; Workshop on Robots at Exhibitions*, Lausanne, Switzerland, 2002.

[5] B. Graf, M. Hans, and R. D. Schraft, *Care-O-bot II—Development of a Next Generation Robotic Home Assistant*, *Autonomous Robots*, vol. 16, no. 2, pp. 193–205, 2004.

[6] B. J. A. Kröse, *et al.*, "Lino, the User-Interface Robot," in *European Symposium on Ambient Intelligence (EUSAI)*, Veldhoven, Netherlands, 2003, pp. 264–274.

[7] "The Cognitive Robot Companion," 2004, http://www.cogniron.org.

[8] J. K. Rosenblatt, "DAMN: A Distributed Architecture for Mobile Navigation," in *Proc. AAAI Spring Symp. on Lessons Learned from Implemented Software Architectures for Physical Agents*. Stanford, CA: AAAI/MIT Press, 1995, pp. 167–178.

[9] È. Coste-Manière and R. G. Simmons, "Architecture, the Backbone of Robotic Systems," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 1, San Franciso, CA, 2000, pp. 67–72.

[10] M. Kleinehagenbrock, J. Fritsch, and G. Sagerer, "Supporting Advanced Interaction Capabilities on a Mobile Robot with a Flexible Control System," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 3, Sendai, Japan, 2004, pp. 3649–3655.

[11] J. Fritsch, *et al.*, *Multi-Modal Anchoring for Human-Robot-Interaction*, *Robotics and Autonomous Systems; Special issue on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, vol. 43, no. 2–3, pp. 133–147, 2003.

[12] G. A. Fink, N. Jungclaus, F. Kummert, H. Ritter, and G. Sagerer, "A Distributed System for Integrated Speech and Image Understanding," in *Int. Symp. on Artificial Intelligence*, Cancun, Mexico, 1996, pp. 117–126.

[13] S. Wrede, J. Fritsch, C. Bauckhage, and G. Sagerer, "An XML Based Framework for Cognitive Vision Architectures," in *Proc. Int. Conf. on Pattern Recognition*, vol. 1, Cambridge, UK, 2004, pp. 757–760.

[14] R. Bischoff, "Recent Advances in the Development of the Humanoid Service Robot HERMES," in *Proc. European Workshop and Masterclass on Advanced Robotics Systems Development (EUREL)*, vol. 1, Manchester, UK, 2000, pp. 125–134.

[15] K. Kim, Y. Matsusaka, and T. Kobayashi, "Inter-Module Cooperation Architecture for Interactive Robot," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 3, Lausanne, Switzerland, 2002, pp. 2286–2291.

[16] B. Hayes-Roth, *A Blackboard Architecture for Control*, *Artificial Intelligence*, vol. 26, no. 3, pp. 251–321, 1985.

[17] M. Lindström, A. Orebäck, and H. I. Christensen, "BERRA: A Research Architecture for Service Robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 4, San Francisco, CA, 2000, pp. 3278–3283.

[18] F. F. Ingrand and F. Py, "An Execution Control System for Autonomous Robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 2, Washington DC, USA, 2002, pp. 1333–1338.

[19] G. Kim, W. Chung, M. Kim, and C. Lee, "Tripodal Schematic Design of the Control Architecture for the Service Robot PSR," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 2, Taipei, Taiwan, 2003, pp. 2792–2797.

[20] G. Kim, W. Chung, M. Kim, and C. Lee, "Implementation of Multi-Functional Service Robots Using Tripodal Schematic Control Architecture," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 4, New Orleans, LA, 2004, pp. 4005–4010.

[21] B. P. Gerkey, R. T. Vaughan, and A. Howard, "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems," in *Proc. Int. Conf. on Advanced Robotics*, Coimbra, Portugal, 2003, pp. 317–323.

[22] "Sony's OPEN-R SDK," http://openr.aibo.com.

[23] M. J. Matarić, *Learning in Behavior-Based Multi-Robot Systems: Policies, Models, and Other Agents*, *Cognitive Systems Research; Special issue on Multi-disciplinary studies of multi-agent learning*, vol. 2, no. 1, pp. 81–93, 2001.

[24] S. Knoop, S. Vacek, R. Zöllner, C. Au, and R. Dillmann, "A CORBA-Based Distributed Software Architecture for Control of Service Robots," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 3, Sendai, Japan, 2004, pp. 3656–3661.

[25] P. Kiatisevi, V. Ampornaramveth, and H. Ueno, "A Distributed Architecture for Knowledge-Based Interactive Robots," in *Proc. Int. Conf. on Information Technology for Application (ICITA)*, Harbin, China, 2004, pp. 256–261.

[26] D. C. Schmidt and S. Vinoski, "Object Interconnections: CORBA and XML, Part 1: Versioning," 2003, http://www.cs.wustl.edu/~schmidt/report-doc.html.

[27] "XCF at Sourceforge.Net," 2004, http://xcf.sourceforge.net.

[28] "The Internet Communications Engine, ZeroC Inc." 2003, http://www.zeroc.com/ice.html.

[29] D. C. Schmidt, M. Stal, H. Rohnert, and F. Buschmann, *Pattern-Oriented Software Architecture*. John Wiley & Sons Ltd., 2000, vol. 2: Patterns for Concurrent and Networked Objects.

[30] A. S. Tanenbaum and M. van Steen, *Distributed Systems: Principles and Paradigms*, 1st ed. Prentice Hall, 2002.

[31] S. Wrede, M. Hanheide, C. Bauckhage, and G. Sagerer, "An Active Memory as a Model for Information Fusion," in *Proc. Int. Conf. on Information Fusion*, vol. 1, Stockholm, Sweden, 2004, pp. 198–205.

[32] "Berkely DB XML, Sleepycat Software," 2004, http://www.sleepycat.com/products/xml.shtml.

[33] E. Gat, "On Three-Layer Architectures," in *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems*. Cambridge, MA: MIT Press, 1998, ch. 8, pp. 195–210.

[34] E. Gat, "Integrating Planning and Reacting in a Heterogenous Asynchronous Architecture for Controlling Real-World Mobile Robots," in *Proc. Nat. Conf. on Artificial Intelligence (AAAI)*. San Jose, CA: AAAI/MIT Press, 1992, pp. 809–815.

[35] A. Haasch, *et al.*, "BIRON – The Bielefeld Robot Companion," in *Proc. Int. Workshop on Advances in Service Robotics*. Stuttgart, Germany: Fraunhofer IRB Verlag, 2004, pp. 27–32.

[36] S. Lang, *et al.*, "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: ACM, 2003, pp. 28–35.

[37] N. Hofemann, J. Fritsch, and G. Sagerer, "Recognition of Deictic Gestures with Context," in *Pattern Recognition; 26th DAGM Symposium, Tübingen, Germany, Aug./Sep. 2004. Proceedings*. Heidelberg, Germany: Springer-Verlag, 2004, vol. 3175, pp. 334–341.

[38] R. J. Firby, "Adaptive Execution in Dynamic Domains," Ph.D. dissertation, Yale University, New Haven, CT, 1989.

[39] "Information Society Technologies Event (IST Event)," Nov. 2004, http://europa.eu.int/information_society/istevent/2004/index_en.htm.

BIRON – The Bielefeld Robot Companion

A. Haasch, S. Hohenner, S. Hüwel, M. Kleinehagenbrock, S. Lang, I. Toptsis,
G. A. Fink, J. Fritsch, B. Wrede, and G. Sagerer
Bielefeld University, Faculty of Technology, 33594 Bielefeld, Germany
Email: ahaasch@TechFak.Uni-Bielefeld.DE

Abstract

In the recent past, service robots that are able to interact with humans in a natural way have become increasingly popular. A special kind of service robots that are designed for personal use at home are the so-called robot companions. They are expected to communicate with non-expert users in natural and intuitive way. For such natural interactions with humans the robot has to detect communication partners and focus its attention on them. Moreover, the companion has to be able to understand speech and gestures of a user and to carry out dialogs in order to get instructed, i.e., introduced to its environment. We address these problems by presenting the current state of our mobile robot BIRON, the Bielefeld Robot Companion.

Keywords: *human-robot interaction, robot companion*

1 Introduction

The development of cognitive robots serving humans as assistants or companions is currently an active research field. In order to be accepted as a communication partner by non-expert users such robot companions must exhibit a human-like communicative behavior. This raises problems related to the sensors used for observing the environment, the techniques employed for data association, and the cognitive capabilities required for multi-modal interaction with humans.

A robot companion will generally be acting in an unstructured environment, such as an office or a private home, with people roaming around. Since it is not desirable to rely on pervasive sensor technology distributed throughout the environment, the robot companion needs to carry all sensing devices on board.

The "field of view" of these sensors will, however, always be limited and their individual capabilities might not be sufficient for robustly interacting with humans. Thus, it is necessary to combine uni-modal processing results in a multi-modal data-association framework. This method increases both reliability in case of occlusions and robustness against processing errors within a single modality.

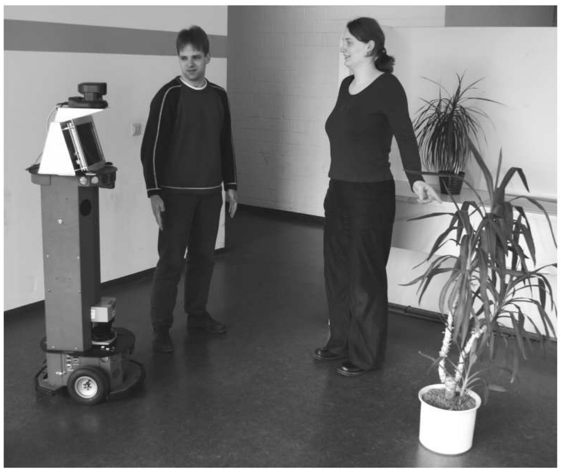


Figure 1: A typical interaction with BIRON.

At the cognitive level a robot companion needs to be able to detect humans and to be aware when a person wants to interact with the robot. For an engagement in a dialog the robot needs to focus its attention on the communication partner and maintain mutual attention throughout the dialog by showing appropriate feedback to the human. For the dialog itself the most important modality is spoken language, which can be complemented by other modalities used in natural communication.

In the envisioned scenario human communication partners can not be expected to wear special equipment, such as a close-talking microphone or data-gloves. Therefore, the multi-modal interaction acts produced by the human must be recognized with the limited sensor capabilities onboard the mobile robot platform alone. Given a semantic interpretation of those multi-modal "utterances" and a symbolic description of the observed scene, appropriate verbal or physical actions of the robot companion can be determined by employing a multi-modal interaction model and strategy.

[1]This work has been supported by the European Union within the 'Cognitive Robot Companion' (COGNIRON) project (FP6-IST-002020) and by the German Research Foundation within the Collaborative Research Center 'Situated Artificial Communicators' as well as the Graduate Programs 'Task Oriented Communication' and 'Strategies and Optimization of Behavior'.

In E. Prassler, G. Lawitzky, P. Fiorini, and M. Haegele, editors, Proc. Int. Workshop on Advances in Service Robotics, pages 27-32, Stuttgart, Germany, May 2004. Fraunhofer IRB Verlag.

In this paper we will present the current state in the development of BIRON, the Bielefeld Robot Companion which is a modified PeopleBot from ActivMedia equipped with a pan-tilt camera, a pair of microphones, and a laser range finder (for details see [4]). Our goal is to use BIRON in the so-called *home-tour* scenario. Here, the basic idea is that a human introduces to a newly purchased robot all the objects and places in a private home relevant for later interaction. Figure 1 shows a typical interaction scene where a user gains the robot's attention in order to engage in a dialog.

2 Related Work

The most advanced examples of robots realizing complex multi-modal human-robot interfaces are *SIG* [13] and *ROBITA* [12]. While only ROBITA is a truly mobile system both robots have a humanoid torso with cameras and microphones embedded in the robot's "head". Both use a combination of visual face recognition and sound source localization for the detection of a potential communication partner. SIG's focus of attention is directed towards the person currently speaking that is either approaching the robot or standing close to it. In addition to the detection of talking people, ROBITA is able to determine the addressee of spoken utterances.

There are also several complete service robot systems that integrate capabilities for human-robot interaction. For example, Care-O-bot II [7] is a multi-functional robot assistant for housekeeping and home care, designed to be used by elderly people. It receives input from the user via speech and touch screen. Although the system also produces speech output, it can not carry out natural dialogs with the user. Lino [8] serves as user interface to intelligent homes. It perceives persons by processing visual and auditory information. Since the robot operates in an intelligent environment it makes use of external information sources. The humanoid service robot *HERMES* [3] can be instructed for fetch-and-carry tasks, and it was also adopted as museum tour guide. It integrates visual, tactile, and auditory data to carry out dialogs in a natural and intuitive way, but can only interact with single persons. Jijo-2 [2] is intended to perform tasks in an office environment, such as guiding visitors or delivering messages. It uses data coming from a microphone array and a pan-tilt camera to perceive persons, but a person is only focused after it says "Hello" to the robot.

3 Overall system architecture

Since interaction with the user is the basic functionality of a robot companion, the integration of interaction components into the architecture is a crucial factor. We propose to use a special control component, the so-called *execution supervisor*, which is located centrally in the robot's architecture. We based our robot control system (depicted in Fig. 2) on a three-layer architecture [6] which consists of three components: a reactive feedback control mechanism, a reactive plan execution mechanism, and a mechanism for performing deliberative computations.

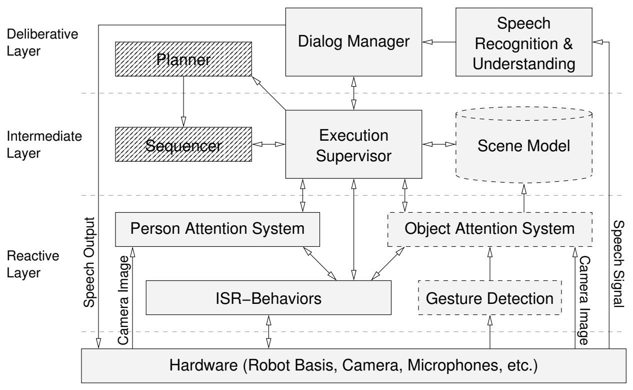


Figure 2: Overview of the BIRON architecture (implemented modules are drawn with solid lines, modules under development with dashed lines).

The execution supervisor, which is the most important component in the architecture, represents the reactive plan execution mechanism. It controls the operations of the modules responsible for deliberative computations rather than vice versa. This is contrary to most hybrid architectures where a deliberator continuously generates plans and the reactive plan execution mechanism just has to assure that a plan is executed until a new plan is received. To continuously control the overall system the execution supervisor performs only computations that take a short time relative to the rate of environmental change perceived by the reactive control mechanism.

While the execution supervisor is located in the intermediate layer of the architecture, the dialog manager is part of the deliberative layer. It is responsible for carrying out dialogs to receive instructions given by a human interaction partner. The dialog manager is capable of managing interaction problems and resolving ambiguities by consulting the user (see section 6). It receives input from the speech understanding system which is also located on the topmost layer (see section 5) and sends valid instructions to the execution supervisor.

The person attention system represents the reactive feedback control mechanism and is therefore located on the reactive layer (see section 4). However, the person attention system does not directly control the robot's hardware. This is done by the ISR software [1]. A parameterization of the attention system leads to the construction of an appropriate network of behaviors inside ISR which then controls the robot's movements.

In addition to the person attention system we are currently developing an object attention system for the reactive layer. The execution supervisor can shift control of the robot from the person attention system to the object attention system in order to focus objects referred to by the user. The object attention will be supported by a gesture detection module which recognizes deictic gestures. Combining spoken instructions and a deictic gesture allows the object attention system to control the robot and the camera in order to acquire visual information of a referenced object. This information will be sent to the scene model in the intermediate layer.

The scene model will store information about objects introduced to the robot for later interactions. This information includes attributes like position, size, and visual information of objects provided by the object attention module. Additional information given by the user is stored in the scene model as well, e.g., a phrase like “This is my coffee cup” indicates owner and use of a learned object.

The deliberative layer can be complemented by a component which integrates planning capabilites. This planner is responsible for generating plans for navigation tasks, but can be extended to provide additional planning capabilities which could be necessary for autonomous actions without the human. As the execution supervisor can only handle single commands, a sequencer on the intermediate layer is responsible for decomposing plans provided by the planner. However, in this paper we will focus on the interaction capabilities of the robot.

4 Person Attention System

A robot companion should enable users to engage in an interaction as easily as possible. For this reason the robot has to continuously keep track of all persons in its vicinity and must be able to recognize when a person starts talking to it. Therefore, both acoustic and visual data provided by the on-board sensors have to be taken into account: at first the robot needs to know which person is speaking, then it has to recognize whether the speaker is addressing the robot, i.e., looking at it. On BIRON the necessary data is acquired from a multi-modal person tracking framework which is based on *multi-modal anchoring* [5].

4.1 Multi-Modal Person Tracking

Multi-modal anchoring allows to simultaneously track multiple persons. The framework efficiently integrates data coming from different types of sensors and copes with different spatio-temporal properties of the individual modalities. Person tracking on BIRON is realized using three types of sensors:

- The laser range finder is used to detect humans' legs. Pairs of legs result in a characteristic pattern in range readings and can be easily detected. From detected legs the distance and direction of the person relative to the robot are extracted [5].
- The camera is used to recognize faces and torsos. Currently, the face detection works for faces in frontal view only [9]. A face provides information about the distance and direction of the person with respect to the robot. In addition, the height of a person can be estimated. Furthermore, the clothing of the upper body part of a person (the color of its torso) can be observed by the camera. If a torso is detected, the direction of the person relative to the robot is known [4].
- The stereo microphones are applied to locate sound sources in front of the robot. By incorporating information from the other cues robust speaker localization is possible [9].

Altogether, the combination of depth, visual, and auditory cues allows the robot to robustly track persons in its vicinity.

In a natural situation, persons are usually moving around. Since also the robot itself is mobile, users can not be expected to be located at a predetermined position. In addition, as the sensing capabilities of the robot are limited, e.g., the camera has only a limited field of view, not all persons in the vicinity of the robot can be observed with all sensors at the same time. To solve these problems an attention mechanism is required.

4.2 Attention Mechanism

The attention mechanism has to fulfill two tasks: On the one hand it has to select the person of interest from the set of observed persons. On the other hand it has to control the alignment of the sensors in order to obtain relevant information from the persons in the robot's vicinity.

The attention mechanism is realized by a finite state machine (see Fig. 3). It consists of several states of attention, which differ in the way the robot behaves, i.e., how the pan-tilt unit of the camera or the robot itself is controlled. The states can be divided into two groups representing *bottom-up attention* while searching for a communication partner and *top-down attention* during interaction.

When bottom-up attention is active, no particular person is selected as the robot's communication partner. The selection of the person of interest as well as transitions between different states of attention solely depend on information provided by the person tracking component. For selecting a person of interest, the observed persons are divided into three categories with increasing degree of relevance. The first category consists of persons that are not speaking. The second category comprises all persons that are speaking, but at the same time are either not looking at the robot or the corresponding decision is not possible,

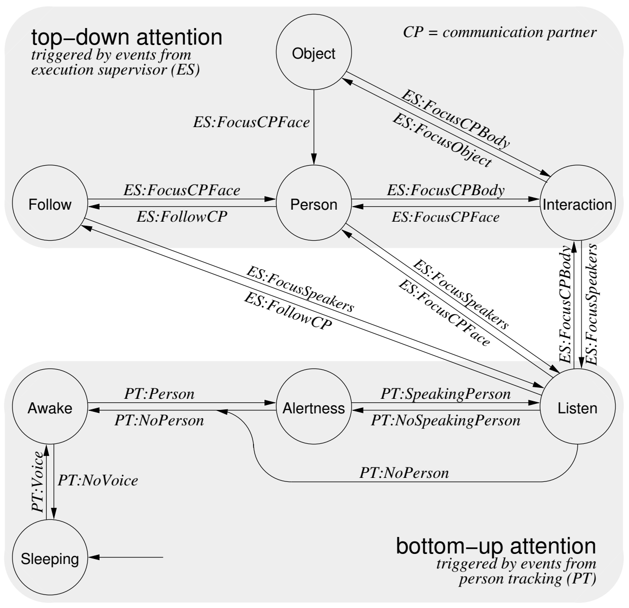


Figure 3: Finite state machine realizing the different behaviors of the person attention mechanism.

since the person is not in the field of view of the camera. Persons assigned to the third category are of most interest to the robot. These persons are speaking and at the same time are looking at the robot. In this case the robot assumes to be addressed and considers the corresponding person to be a potential communication partner. If a person is assigned to this category it is instantly selected and remains selected until the person changes to one of the other categories, e.g., by stopping talking or looking in another direction. If no person has the status of a potential communication partner, the attention mechanism always selects the person that is of most interest, e.g., persons of the second category are selected prior to persons of the first category. If the mechanism has to decide between multiple persons of the same category, it selects the one that for the longest time was not selected. In addition, the mechanism will also switch between persons in order to obtain additional information, e.g., the identities of persons present. For this purpose, a person remains selected only for a limited amount of time, after which it is temporarily blocked for selection, realizing an effect known as inhibition of return.

Top-down attention is activated as soon as the robot starts to interact with a particular person. During interaction the robot's focus of attention remains on this person even if it is not speaking. Here, in contrast to bottom-up attention, transitions between different states of attention are solely triggered by the execution supervisor. The corresponding events sent by the execution supervisor depend on the current state of the dialog.

The behavior of the robot concerning the states of the attention mechanism differs in the way the pan-tilt unit of the camera and the robot itself is controlled. Except for the two states *Sleeping* and *Object* (see Fig. 3) the camera is oriented towards the selected person, primarily towards the user's face, but also towards the torso (*Follow*) in order to robustly track the person while following, or towards the user's hands (*Interaction*) in order to be able to capture deictic gestures. When the attention mechanism is in the state *Listen* or in one of the states of top-down attention, the selected person is likely to speak to the robot. In order to obtain optimal quality of the acoustic signal the robot turns towards the person. Except for the state *Follow* the robot is not moving forward. When the attention mechanism is in the state *Object* the camera is oriented towards a predetermined position in our current implementation. Now we are developing a self-contained object attention mechanism which will replace this state.

5 Speech Recognition and Understanding

Speech is the most important modality for a multimodal dialog. On BIRON there are two major challenges. First, speech recognition has to be performed on distant speech data recorded by the two on-board microphones. And second, speech understanding has to deal with spontaneous speech phenomena.

The recognition of distant speech with two (or more) microphones can be achieved by reconstructing a single channel representation of the speech originating from a known location on the basis of the different channels recorded by the microphones. This technique is known as beam-forming [10] and calculates a weighted average of the individual channels taking into account the estimated time delay. For recognizing distant speech we calculate this single channel reconstruction by applying beam-forming in the log-spectral domain. This method produces better results on the data recorded via the microphones on BIRON than beam-forming in the time or spectral domain.

The activation of speech recognition is controlled by the attention mechanism. Only if a tracked person is speaking and looking at the robot at the same time, speech recognition and understanding takes place. Since the position of the speaker relative to the robot is known from the person tracking component, the time delay can be estimated and taken into account for the beam-forming process. However, since noise and speech from interfering talkers standing at different positions can only be suppressed to some extent by beam-forming, the recognition quality will never reach the one obtained with a close-talking microphone.

Besides this problem of the speech recognition system the speech understanding component has to deal with spontaneous speech phenomena in dialogs between a user and the robot. For example, large pauses and incomplete utterances can occur in such task oriented and embodied communication. However, missing information in an utterance can often be acquired from the scene. For example the utterance "Look at this" and a pointing gesture to the table concludes to the meaning "Look at the table". Moreover, fast extraction of semantic information is important for achieving adequate response times.

We obtain fast and robust speech processing by combining the speech understanding component with the speech recognition system. For this purpose, we integrate a robust LR(1)-parser into the speech recognizer as proposed in [15]. Besides, we use a semantic-based grammar which is used to extract instructions and corresponding information from the speech input. A semantic interpreter forms the results of the parser into frame-based XML-structures and transfers them to the dialog manager (see section 6). Hints in the utterances about gestures are also incorporated. For our purpose, we consider co-verbal gestures only. An utterance as "This flower at the window" is transformed to the structure in Figure 4. The object attention system is intended to use this information in order to detect a specified object. Thus, this approach supports the object attention system and helps to resolve potential ambiguities.

```
<SPEECH>
   <TIMESTAMP val = "4071866790" />
   <OBJECT type = "plant" >
      <GESTURE val = "probably_pointing" />
      <TITLE name = "flower" />
      <POSITION >
         <RELATION name = "at" />
         <TITLE name = "window" />
      </POSITION >
   </OBJECT >
</SPEECH>
```

Figure 4: Representation of *"This flower at the window"*

6 Dialog Manager

The model of the dialog manager is based on a set of *finite state machines* (FSM), where each FSM represents a specific dialog. The FSMs are extended with the ability of recursive activation of other FSMs and the execution of an action in each state. Actions that can be taken in certain states are specified in the *policy* of the dialog manager. These actions include the generation of speech output and sending events like orders and requests to the execution supervisor. The dialog *strategy* is based on the so-called *slot-filling* method [14]. A slot is an information item for which a value is required. The status of a slot can be empty, filled with an attribute, or in case of a binary entry be *true* or *false*. For every FSM a set of slots is available, which are organized in a so-called dialog *frame*. Every different status combination of the slots in a frame defines a state in the corresponding FSM of the model. The task of the dialog manager is to fill enough slots to meet the current dialog goal, which is defined as a goal state in the corresponding FSM. The slots are filled with information coming from the user and other components of the robot system. This procedure can be viewed as a quantization of a user utterance into required information items.

The dialog management is event-based, where switching between the dialog states is not done by following a transition in the model, but depends on the status composition of all slots in the dialog frame. Several input events like user utterances or information from other components of the robot system change the status of the slots. In an ongoing dialog, the dialog manager compares the slots in the newly updated dialog frame with those in the FSM to find the model's new current state. Thereby, slots are compared only by their status and not by their content. After executing an action, which is determined by a lookup in the dialog policy, the dialog manager waits for new input from the execution supervisor or the speech understanding system.

The slot-filling technique alone is not powerful enough to support the complex interaction scenarios in robot domains [11]. The user intentions are not predictable in such cases. To overcome this limitation, we designed the dialog in a modular way and divided each dialog into a set of sub-dialogs. Each sub-dialog is responsible for a task and is modeled as a separate FSM. This FSM has a goal state which indicates the completion of the current task. The processing of each sub-dialog can be interrupted by another sub-dialog, which makes alternating instruction processing possible. The dialogs are specified using a declarative definition language and encoded in XML in a modular way. This increases the portability of the dialog manager and allows an easier configuration and extension of the defined dialogs.

7 Interaction Capabilities

In the following we describe the interaction capabilities BIRON offers to the user in our current implementation. Initially, the robot observes its environment. If persons are present in the robot's vicinity, it focuses on the most interesting one (cf. section 4). A user can start an interaction by greeting the robot with, e.g., "Hello BIRON". Then, the robot keeps this user in its focus and can not be distracted by other persons talking. Next, the user can ask the robot to follow him to another place in order to introduce it to new objects. While the robot follows a person it tries to maintain a constant distance to the user and informs the person if it moves too fast. When the robot reaches a desired position the user can instruct it to stop. Then, the user can ask the robot to learn new objects. In this case the camera is lowered to also get the hands of the user in the field of view.

When the user points to a position and gives spoken information like “This is my favorite cup”, the object attention system is activated in order to center the referred object. If the user says “Good-bye” to the robot or simply leaves, the robot assumes that the current interaction is completed and looks around for new potential communication partners.

8 Summary

In this paper we presented an overview of the robot companion BIRON whose target application is the home-tour scenario. Its natural interaction capabilities are based on a person attention system, a speech recognition and understanding component, and a dialog manager. These components are integrated in a hybrid architecture which is controlled by a central execution supervisor. The architecture’s modular design easily allows modifications on the robot companion’s skills by replacing and adding new components. Current work focuses on switching to a powerful communication framework [16] and integrating an object attention system for associating gestures with visual features of objects.

References

[1] M. Andersson, A. Orebäck, M. Lindstrom, and H. I. Christensen. ISR: An intelligent service robot. In H. I. Christensen, H. Bunke, and H. Noltmeier, editors, *Sensor Based Intelligent Robots; International Workshop Dagstuhl Castle, Germany, September/October 1998, Selected Papers*, volume 1724 of *Lecture Notes in Computer Science*, pages 287–310. Springer, New York, 1999.

[2] H. Asoh, Y. Motomura, F. Asano, I. Hara, S. Hayamizu, K. Itou, T. Kurita, T. Matsui, N. Vlassis, R. Bunschoten, and B. Kröse. Jijo-2: An office robot that communicates and learns. *IEEE Intelligent Systems*, 16(5):46–55, 2001.

[3] R. Bischoff and V. Graefe. Demonstrating the humanoid robot *HERMES* at an exhibition: A long-term dependability test. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems; Workshop on Robots at Exhibitions*, Lausanne, Switzerland, 2002.

[4] J. Fritsch, M. Kleinehagenbrock, S. Lang, G. A. Fink, and G. Sagerer. Audiovisual person tracking with a mobile robot. In *Proc. Int. Conf. on Intelligent Autonomous Systems*, pages 898–906. IOS Press, 2004.

[5] J. Fritsch, M. Kleinehagenbrock, S. Lang, T. Plötz, G. A. Fink, and G. Sagerer. Multi-modal anchoring for human-robot-interaction. *Robotics and Autonomous Systems, Special issue on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, 43(2–3):133–147, 2003.

[6] E. Gat. On three-layer architectures. In D. Kortenkamp, R. P. Bonasso, and R. Murphy, editors, *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems*, chapter 8, pages 195–210. MIT Press, Cambridge, MA, 1998.

[7] B. Graf, M. Hans, and R. D. Schraft. Care-O-bot II—Development of a next generation robotic home assistant. *Autonomous Robots*, 16(2):193–205, 2004.

[8] B. J. A. Kröse, J. M. Porta, A. J. N. van Breemen, K. Crucq, M. Nuttin, and E. Demeester. Lino, the user-interface robot. In *European Symposium on Ambient Intelligence (EUSAI)*, pages 264–274, 2003.

[9] S. Lang, M. Kleinehagenbrock, S. Hohenner, J. Fritsch, G. A. Fink, and G. Sagerer. Providing the basis for human-robot-interaction: A multi-modal attention system for a mobile robot. In *Proc. Int. Conf. on Multimodal Interfaces*, pages 28–35. ACM, 2003.

[10] S. J. Leese. Microphone arrays. In G. M. Davis, editor, *Noise Reduction in Speech Applications*, pages 179–197. CRC Press, Boca Raton, London, New York, Washington D.C., 2002.

[11] O. Lemon, A. Bracy, A. Gruenstein, and S. Peters. The WITAS multi-modal dialogue system I. In *Proc. European Conf. on Speech Communication and Technology*, pages 1559–1562, Aalborg, Denmark, 2001.

[12] Y. Matsusaka, T. Tojo, and T. Kobayashi. Conversation robot participating in group conversation. *IEICE Trans. on Information and System*, E86-D(1):26–36, 2003.

[13] H. G. Okuno, K. Nakadai, and H. Kitano. Social interaction of humanoid robot based on audio-visual tracking. In *Proc. Int. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, Cairns, Australia, 2002. Lecture Notes in Artificial Intelligence, Springer.

[14] B. Souvignier, A.Kellner, B. Rueber, H. Schramm, and F. Seide. The thoughtful elephant: Strategies for spoken dialog systems. In *IEEE Trans. on Speech and Audio Processing*, volume 8, pages 51–62, 2000.

[15] S. Wachsmuth, G. A. Fink, and G. Sagerer. Integration of parsing and incremental speech recognition. In *Proc. European Conf. on Signal Processing*, volume 1, pages 371–375, Rhodes, 1998.

[16] S. Wrede, J. Fritsch, C. Bauckhage, and G. Sagerer. An XML based framework for cognitive vision architectures. In *Proc. Int. Conf. on Pattern Recognition*, Cambridge, UK, 2004. to appear.

Supporting Advanced Interaction Capabilities on a Mobile Robot with a Flexible Control System

M. Kleinehagenbrock, J. Fritsch, and G. Sagerer
Applied Computer Science Group
Faculty of Technology, Bielefeld University
33594 Bielefeld, Germany
Email: {mkleineh, jannik, sagerer}@techfak.uni-bielefeld.de

Abstract— **Building a mobile service robot for home and office environments that incorporates skilled interaction capabilities is a challenging task. The control system has to consider various demands: Firstly it has to manage unstructured and dynamic environments. Secondly, as humans are around, aspects of safety are of particular importance. This implies that the system has to be highly reactive. Thirdly, the robot also has to be capable of carrying out dialogs to be taught or instructed. Altogether, this requires a highly integrated control framework. In this paper we present an agent-based architecture for our mobile robot BIRON in order to realize sophisticated human-robot interaction. The architecture is build in a modular fashion and controlled by a central execution supervisor using an event queue to handle asynchronous events. This execution supervisor contains an augmented finite state machine which is specified in XML and thus is highly generic. Similarly, the communication between all modules is based on XML. The overall system, therefore, is easily maintainable and extensible, as the architecture's design allows us to add new modules to the system without requiring major modifications on existing components.**

I. Introduction

Service robots have received increased attention in research in the last years. Various systems have been developed, starting from tour guides (e.g., [8], [28]) across messenger robots (e.g., [29]) up to home care robots (e.g., [16], [26]). Recently, another application scenario has become increasingly interesting: the so-called *home tour scenario*. Here, the basic idea is that a human introduces a newly bought robot all the objects and places in a private home relevant for later interaction between the human and the robot.

Our goal is to provide BIRON – the Bielefeld Robot Companion – with interaction capabilities required for the home tour scenario. Because the mobile robot is instructed by humans the ability to detect and track persons in the environment is an important prerequisite for the interaction. Therefore, we built a multi-modal anchoring framework [11] which allows the robot to track multiple persons in its vicinity in unrestricted environments using visual, auditory, and depth cues. Based on our multi-modal anchoring approach we developed a person attention system [21] which allows the robot to focus its attention on potential communication partners by directing its sensors towards these persons.

*The work described in this paper was supported by the EU Integrated Project COGNIRON ("The Cognitive Companion") and funded by the European Commission Division FP6-IST Future and Emerging Technologies under Contract FP6-002020 and by the German Research Foundation within the Graduate Program "Strategies and Optimization of Behavior".

In order to communicate with the robot a dialog control module is needed for the system. Dialogs allow a user to change the robots behavior in order to fulfill certain tasks. While the robot's person attention system is a reactive component, the dialog control module introduces high-level control to the system. For the combination of these two different kinds of control an overall system architecture is necessary. As already mentioned, our focus lies not only on developing a corresponding concept, but also on actually realizing a robot which has advanced interaction capabilities. Therefore, the design of an appropriate architecture is also influenced by technical aspects like, e.g., ease of modification. This paper presents our architecture concept and how it is applied to our robot.

The rest of the paper is organized as follows: In section II we summarize the four basic methodologies of robot control architectures. Then, systems with human-robot interaction capabilites are reviewed in section III. Next, we introduce our concept of an agent-based architecture suitable for integrating advanced human-robot interaction capabilities in section IV. This concept is applied to our robot described in section V and the resulting system is presented in section VI. In section VII we take a closer look at the central supervising component of our architecture, the execution supervisor. In section VIII we present the current functionality of the overall system. The paper concludes with a short summary in section IX.

II. Robot Architectures

The task of a robot software architecture is to structure control, which is the process of acquiring information about the environment through the robot's sensors, processing acquired information as required in order to decide about how to act, and executing actions in the environment.

In the past decades, four basic classes of control emerged for robot architectures: Reactive, deliberative, hybrid, and behavior-based control. Matarić briefly summarized their methodology as follows [25]:

- Deliberative Control: Think hard, then act.
- Reactive Control: Don't think, act.
- Hybrid Control: Think and act independently, in parallel.
- Behavior-Based Control: Think the way you act.

In the following we will give a short introduction on these different control methodologies.

A. Deliberative control

Deliberative control which is sometimes also called *planner-based control* uses *sense-plan-act* decomposition to construct intelligent systems [27]. This methodology emerged from traditional AI which relies on robust, action independent representations of the world. In deliberative control *sense*, *plan*, and *act* are carried out sequentially. In the first step all sensor information is acquired. This is then compared to the internally stored knowledge in order to reason about what actions to perform. Reasoning is typically achieved by planning where various sequences of actions are analyzed to find the most promising sequence for reaching a specific goal. Finally, the actions of the most promising plan are executed to reach the goal.

However, for embodied systems like mobile robots the search space is typically large and only partially observable making planning computationally expensive. As sensors have certain limitations, the environment is usually noisy and uncertain. This can make it problematic for a robot to reach its goal. If dynamic changes in the environment have also to be considered, re-planning might be necessary very often.

B. Reactive control

While deliberative control can provide high-level intelligence by planning, it normally fails in an incorrectly represented or unknown environment. Therefore, reactive control systems have been proposed which do not use a world model and refer to sensor data only, following the slogan *"The world is its own best model"* [7]. This shift from *sense-plan-act* to reactive systems is often inspired and accompanied by the interest in how biological systems manifest intelligent behavior. As a result, the reactive methodology advocates more direct connections between sensors and actuators [3]. This leads to a tradeoff against complexity of reasoning and in favor of fast reaction time, allowing the robot to quickly respond to changing and unstructured environments. A well known example for reactive control is given by the Subsumption architecture [6].

However, reactive control is unable to reason about the future, as it is based on the system's current state without any lookahead or considering hypothetical states of the world. Reactive control also encodes neither goals nor sequences of possible actions. Moreover, reactive systems lack run-time flexibility as they have no memory and thus have no ability to learn over time.

C. Hybrid control

In order to overcome the restrictions of both deliberative and reactive control the idea of hybrid control is to combine their best aspects. One of the first hybrid architectures is AuRA [1], combining a hierarchical planner and a control system based on motor schemas [2]. In general, hybrid control aims to combine the real-time response of reactivity with the rationality of deliberation. The strategy is, on the one hand, to enable a robot to exhibit ongoing behavior by applying a reactive component which provides quick responses in dynamic environments. On the other hand, the capability to perform complex tasks is supplied by a deliberative component which generates corresponding plans. These components run independently, but reactive control has to override deliberative control when the environment presents a sudden challenge. Contrary, deliberative control has to instruct reactive control in order to direct the robot to follow efficient or even optimal strategies for reaching a specific goal.

The interaction between the two control components needs sophisticated scheduling, because reactive control mainly processes sensory signals and thus operates on a short time scale while deliberative control operates on a longer time scale. Therefore, for the interaction between reactive and deliberative components most researchers propose an intermediary component, whose design is typically the greatest challenge of hybrid systems [25]. Resulting systems are usually called *three-layer systems* and consist of a reactive, an intermediate, and a deliberative layer.

D. Behavior-based control

Behavior-based control which is also called *reactive subsumption-style control* is inspired from biology for its design of situated and embodied systems. Behavior-based systems received their name from their basic components, called *behaviors*, which are observable patterns of activity emerging from interactions between the robot and its environment [25]. Behavior-based control should not be mistaken with purely reactive control. It differs from the latter by the fact that complementary and contradictory actions are generated by the system in parallel. These actions are combined by one or more arbiters in order to obtain a final action to be taken by the robot. Behaviors can involve establishing internal representations and thus enabling deliberation and learning. Some examples on behavior-based robots are given in [24].

In behavior-based systems internal representations are stored by behaviors locally and therefore information is distributed throughout the behavior network. Thus, planning is done by a network of communicating behaviors, rather than by a centralized planner. This circumstance has to be considered when computational and performance issues are important. The way arbiters fuse the output of behaviors is determined at design time and different weights are usually put on the output of different behaviors in order to increase flexibility, but such adjustments are rarely trivial. Moreover, large networks are difficult to maintain, as behaviors can become unnecessary or not desired any more.

E. Comparing the methodologies

The appropriateness and performance of a control architecture for a mobile robot very much depends on the intended application. Moreover, the decision which control methodology to choose is also robot specific. Purely deliberative and reactive control are mostly used for very special purposes, as they only work well within a controlled domain or for low-level tasks. In contrast, hybrid and behavior-based systems have more expressive and computational capabilities which are very similar to each other. Both of them can store representations and make predictions about the future, but each does it differently. Hybrid systems are prevalent in single robot domains as hybrid control scales well for the needs of service and humanoid robots. Contrary, in multi-robot domains behavior-based systems are more usual because the adaptability of behavior-based control allows the robots to show desired group behavior in a flexible way. Behavior-based control is also often used for pet robots.

III. Systems incorporating HRI

There are several mobile robot systems that integrate capabilities for human-robot interaction (HRI) in their architecture. For example, Care-O-bot [16] is a multi-functional robot assistant for housekeeping and home care, to be used by elderly people. Its architecture is hybrid and uses JAM [17], a Belief-Desire-Intention (BDI) framework, for plan execution. Lino [20] serves as user interface to an intelligent environment or home. Its reasoning mechanism is also based on a BDI architecture. The humanoid service robot *HERMES* [5] can be instructed for fetch-and-carry tasks, but it was also adopted as museum tour guide. Although its system's core is behavior-based, the robot has a situation-oriented deliberative component to pursue long-term goals. Jijo-2 [4] is intended to perform tasks in an office environment, such as guiding visitors or delivering messages. Its system includes a reactive and an integrator layer, also forming a hybrid architecture. With respect to their architecture, most systems incorporating human-robot interaction can be classified as hybrid. BDI is often used for modeling the deliberative component of a robot system, but then a reactive counterpart is generally needed, as reactivity in BDI is restricted by the rate at which individual actions are executed.

As interaction with a user is essential for our work, the integration of a corresponding component is a crucial factor. We propose to connect it to a special control component located centrally in the architecture. Concerning this kind of central control, there are some architectures bearing similarities to ours. An example is the LAAS architecture [18] running on the robot Diligent. It contains a central component, the so-called supervisor/executive, which coordinates data coming from the robot system and the planner, and commands from the operator. The system does not include human-robot interaction capabilites as the focus is on the execution of valid and safe commands which are verified in hard real time. The BERRA architecture [23] is used for diverse Nomad robots. It is applied for fetch-and-carry tasks and as tour guide in the office domain. BERRA is a three-layer architecture, designed to provide scalability and a high degree of flexibility. Human-robot interaction covers mission acquirement only and, therefore, the corresponding input is routed directly into the planner. The robot PSR [19] has also a three-layer architecture which enables the robot to fulfill certain transportation tasks. The overall system configuration is modeled by Petri nets which is advantageous for modeling coordination of parallel processes. However, temporal synchronization is not explicitly modeled and is realized by an ad hoc solution.

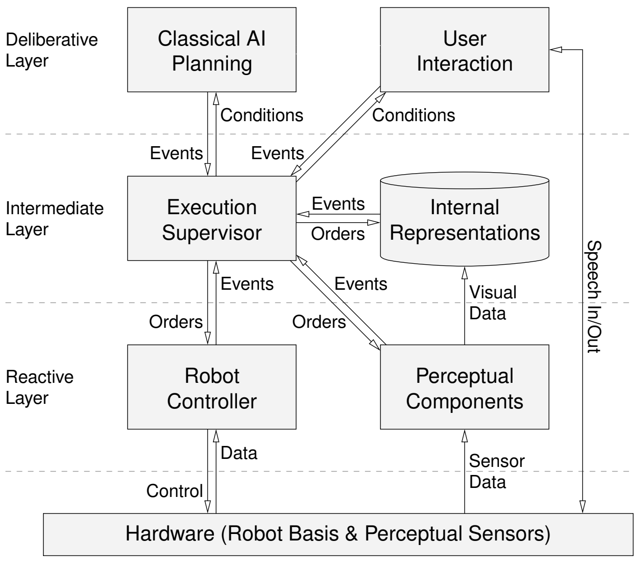


Fig. 1. Conceptual model of the proposed architecture.

IV. Our architecture concept

After reviewing existing systems we decided to develop a concept for a hybrid architecture as it is the most flexible way to organize a system which integrates autonomous control and human-robot interaction capabilities. In principle our system is based on a three-layer architecture [13]. Three-layer architectures consist of three components: a reactive feedback control mechanism, a reactive plan execution mechanism, and a mechanism for performing time-consuming deliberative computations. These components typically run as separate computational processes.

The most important component concerning the structure of the proposed architecture is a central *execution supervisor*, which represents the reactive plan execution mechanism. The functionality of this execution supervisor is similar to the so-called *sequencer* used for ATLANTIS [12] where it coordinates the operations of the modules responsible for deliberative computations rather than vice versa. This is contrary to most hybrid architectures where a deliberator continuously generates plans and the reactive plan execution mechanism just has to make sure that a plan is executed until a new plan is received.

To continuously control the overall system the execution supervisor should only perform computations that take a short time relative to the rate of environmental change perceived by the reactive control mechanism. An overview of the concept of our architecture can be seen in Fig. 1.

While the execution supervisor is located in the intermediate layer, the robot controller represents the reactive feedback control mechanism and is therefore located in the reactive layer. The design of the controller depends on the purpose of the robot and can be based on, e.g., behaviors. The controller has direct access to the hardware and should be parameterizable by the execution supervisor in order to fulfill diverse tasks.

The deliberative layer contains a planner which generates action sequences. Its overall tasks are initiated by a user interaction module which is located in the same layer. The planner receives such tasks via the execution supervisor which assures that the user instructions are consistent with the overall system state. The user interaction module is responsible for carrying out dialogs with human interaction partners in order to acquire all information needed to fulfill a certain task. Therefore, it processes speech input and generates corresponding speech output. Because the user interaction module is directly connected to the execution supervisor, ambiguities which might arise from modules in the reactive layer can also be resolved by dialog. For this purpose corresponding enquiries from the reactive layer are routed through the execution supervisor.

The architecture is complemented by perceptual components in the bottom layer and a module for storing internal representations in the intermediate layer. The perceptual components process sensor data in order to gather information about the environment. This information can then be stored as internal representation together with additional information acquired from a dialog with the user.

The data flow between all modules in the architecture is event-based and every message is coded in XML [30]. This concept is motivated by non-functional aspects: As both, data structures and communication channels, are *name based* the different modules are loosely coupled in a declarative manner and can easily be replaced by others. It also eases modification concerning the data structures used for communication, as XML allows database-like exchange of information. Therefore, this approach allows modules to act as independent agents in the architecture which facilitates simplicity, usability, and flexibility.

The communication is structured as follows: Each module sends *events* containing configuration data for other modules to the execution supervisor. As a result of processing an event, the execution supervisor sends *orders* and *conditions* including needed parameters which are supplied by the event. Orders are sent to all modules in the intermediate and reactive layer in order to reconfigure the system. Conditions are sent to modules in the deliberative layer which inform these modules about the internal state of the overall system. This form of communication reflects the hierarchical structure of the architecture: Orders are sent ‘down’, while conditions are sent ‘up’. A communication example is given in the description of the concrete implementation of our execution supervisor in section VII.

In order to satisfy certain system safety requirements, modules should fail perceivably as in a real world robot failures cannot be excluded. We realized this feature by messages which are initiated by the main loop of each module and sent in fixed intervals to modules being in communication with this module. If a module does not receive these messages anymore, it can determine that the corresponding sender stopped working correctly. In this case corrective actions can be taken to recover from the failure. If recovery is not possible then the robot is at least able to ask the user to call technical support.

V. Robot Hardware

Our architecture concept is implemented on our mobile robot BIRON (Fig. 2). Its hardware platform is a Pioneer PeopleBot from ActivMedia with an on-board PC (Pentium III, 850 MHz) for controlling the motors and the on-board sensors and for sound processing. An additional PC (Pentium III, 500 MHz) inside the robot is used for image processing.

The two PCs running Linux are linked by an 100 Mbit Ethernet LAN and the controller PC is equipped with wireless LAN to enable remote control of the mobile robot. As additional interactive device a 12” touch screen display is provided on the robot.

A pan-tilt color camera (Sony EVI-D31) is mounted on top of the robot at a height of 141 cm for acquiring images of the upper body part of humans interacting with the robot. Two AKG far-field microphones which are usually used for hands free telephony are located at the front of the upper platform at a height of 106 cm, right below the touch screen display. The distance between the microphones is 28.1 cm. A SICK laser range finder is mounted at the front at a height of approximately 30 cm.

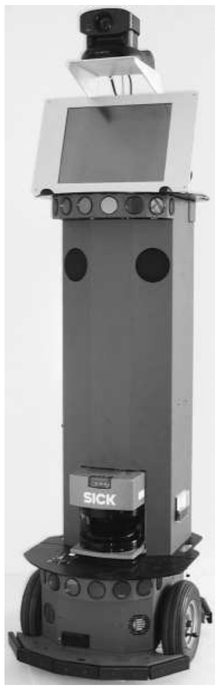


Fig. 2. BIRON.

VI. Architecture of BIRON

In this section we describe how the proposed architecture concept is implemented on our robot interacting with users in the home tour scenario. An overview of the resulting architecture can be seen in Fig. 3.

The main modules in the deliberative layer are the planner and the dialog control agent. The planner is responsible for generating plans for navigational tasks, but it can be extended to provide additional planning capabilities which could be necessary for autonomous actions without the human. However, this is not our focus right now, as such actions are initiated after the robot is fully instructed. The dialog control module is responsible for carrying out dialogs to receive instructions given by a human interaction partner. It is capable of managing interaction problems and resolving ambiguities by consulting the user.

The dialog control is based on a set of finite state machines, each representing a certain sub-dialog. It receives input from the speech understanding system (see [15] for more details on these two components). The dialog control

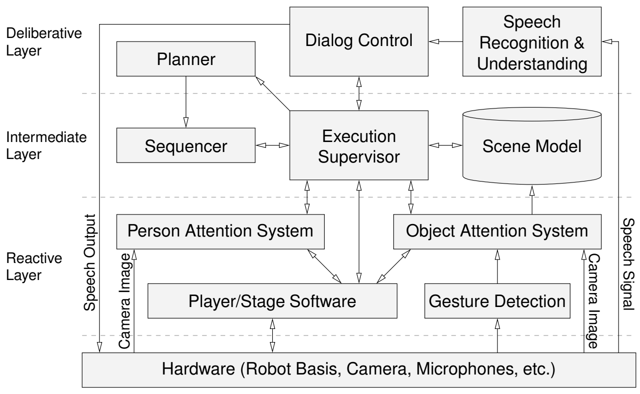


Fig. 3. Actual architecture model implemented on BIRON.

sends valid instructions to the execution supervisor (see section VII for details) which is located in the intermediate layer of our architecture. The sequencer also resides in this layer. As the execution supervisor can only handle single commands, the sequencer is responsible for decomposing plans provided by the planner.

The person attention system [21] represents the main reactive feedback control mechanism and is therefore located in the reactive layer. It detects potential communication partners among persons present in the vicinity of the robot. It is configured by the execution supervisor to show different behaviors, e.g., to look at all people in its surrounding or to track a specific communication partner. However, the person attention system does not directly control the robot's hardware. This is done by the *Player/Stage* software [14]. *Player* provides a clean and simple interface to the robot's sensors and actuators. Even though we currently use this software to control the hardware directly, the controller can easily be replaced by a more complex one which may be based on, e.g., behaviors.

Besides the person attention system an object attention system is located in the reactive layer. The execution supervisor can shift control of the robot from the person attention system to the object attention system in order to focus objects referred to by the user. The object attention is supported by a gesture detection module which recognizes deictic gestures. Combining spoken instructions and a deictic gesture allows the object attention system to acquire visual information of a referenced object. This information is sent to the scene model in the intermediate layer.

The scene model stores information about objects introduced to the robot for later interactions. This information includes attributes like position, size, and visual information of objects provided by the object attention module. Besides, additional information given by the user is stored in the scene model, e.g., a phrase like “This is my coffee cup” indicates owner and use of a learned object.

VII. The execution supervisor

The execution supervisor is the central part of our architecture and is designed to be as generic as possible. In order to achieve this requirement the execution supervisor interprets no data at all: It either configures modules of the system at run time based on received events containing needed parameters, or it routes specific data to modules which are responsible for processing the data. Since all information is carried by XML documents, the execution supervisor even does not need to distinguish between the data structures it receives. However, different data can be stored and concatenated before it is directed to a receiver.

The execution supervisor is controlled by an *augmented finite state machine* (AFSM). As the execution supervisor is not intended to interpret any data, the AFSM is not very complex. Therefore, the structure of the AFSM is also specified in XML. As a consequence, no special-purpose language like, e.g., RAPs [9] is necessary. Defining the AFSM in XML results in a clear representation which allows to quickly restructure or extend the execution supervisor without recompiling it. This concept also allows us to modify the execution supervisor at run time in order to enable it to learn new module interactions over time. As new XML documents can be handled directly by default their integration is straightforward.

The finite state machine is augmented in so far, that with every transition which is executed, a specific *action* is performed. These actions are for configuring the system by emitting two specific types of events: *conditions* and *orders*. In our system orders are sent by the execution supervisor to the scene model, the sequencer, both attention systems, and the *Player* software, while conditions are sent to the dialog control agent and the planner.

Conditions inform the dialog control module about the internal status of the robot system, e.g., information about the current communication partner. Similarly, the planner receives a navigational task in the form of a condition to generate a corresponding plan. In contrast, orders sent to the attention systems contain configuration data to change the behavior of these modules. The *Player* software receives commands for direct control. Orders sent to the scene model cause it to either store or provide data. In the latter case the scene model generates a list of objects that match a corresponding request. It sends this list back to the execution supervisor which routes it to the dialog control. When a plan is executed, the sequencer is queried for new plan steps until the corresponding task is finished.

Besides sending conditions and orders, the execution supervisor also receives data in the form of events from all modules connected to it. These events initiate the generation of conditions and orders. Because the execution supervisor receives events from other modules asynchronously, it uses an *event queue*. All incoming events contain timestamps indicating when they were created. All events are inserted in the queue ordered by their timestamps. The events are handled in turn, starting with the oldest one. Each event corresponds to a transition in the AFSM. Thus, transitions from one state to another can only be triggered by events. When a transition is executed, the corresponding event is deleted from the event queue.

The event queue is also used to synchronize events. For example, a person can only become the robot's current communication partner if the person attention system signals that it has detected a potential communication partner and the dialog control notifies the execution supervisor that it has received a corresponding speech input. Only if both events arrive at the execution supervisor in a certain interval of time, it is assumed that these events belong together. Consequently, the execution supervisor changes to a state which causes the person attention system to focus on the corresponding person.

The execution supervisor also has to consider latencies between incoming events, as all modules run asynchronously. This is realized by a life time entry in the corresponding events which describes how long an event remains valid. Although events could be synchronized by applying them to a transition in combination, we only label transitions with single events and therefore handle them sequentially. This is done, because processing of a particular event may be necessary while the rest of the events needed for synchronization are still missing. To refer to the example above, the system has to distinguish whether a potential communication partner is detected or not, independently from determining that the person is speaking. To retain events in the event queue until other events arrive which must be applied first, some states of the AFSM are grouped in *categories*.

Altogether, there are two cases where events can not be handled directly. An event is not handled if

1) it is out of date, which is indicated by the life time entry. In this case the event is deleted from the event queue and rejected. If the event came from the dialog control agent it is sent back, including the reason for the rejection. This allows the dialog control to communicate the problem to the user.
2) it can not be applied to the current state of the AFSM. If the intended transition belongs to the category the AFSM is currently in, the event is skipped and retained until it can be applied to a state of this category or it is out of date. Otherwise, the event is deleted from the event queue and rejected.

The mechanism described above ensures that certain events from different modules have to arrive in a certain time interval before the execution supervisor changes to a specific state. This reflects that the execution supervisor is in control of the overall system. The dialog control agent can give advices, but if the components of the remaining layers do not provide corresponding information advices are rejected. For example, the robot may perceive instructions from a radio, but it will not start an interaction as no person can be detected. Thus, the synchronization of events received from modules of different layers in the architecture makes the overall system more robust.

The AFSM which is currently used in our system is shown in Fig. 4. The grey bar indicates a state category. Labels in quotation marks denote transitions that are executed by processing events which come from the dialog control agent, while italic labels refer to events received from one of the attention systems. The remaining labels are related to events coming from the scene model.

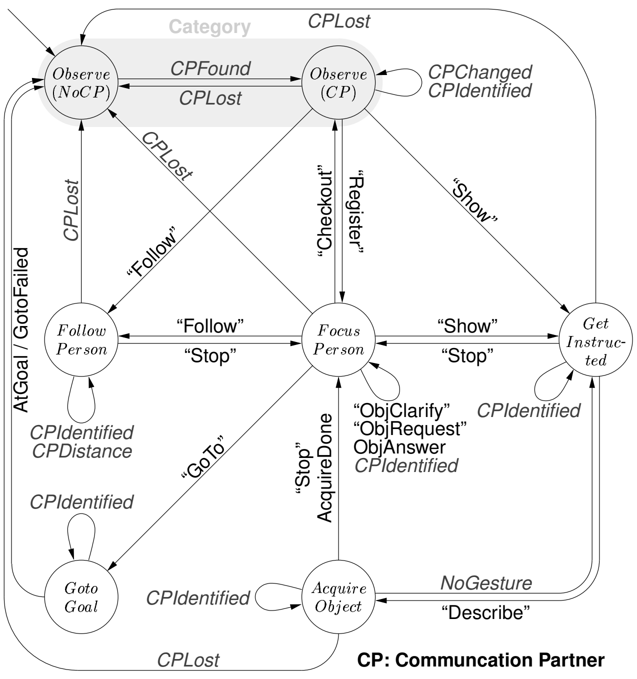


Fig. 4. Augmented finite state machine of the execution supervisor.

Our concept of the execution supervisor also scales up if the system is extended. Generally, only one more state is needed when a new functionality or module is added.

VIII. Interaction capabilities

In the following we describe the interaction capabilities BIRON offers to the user in our current implementation. Initially, the robot observes its environment. If persons are present in the robot's vicinity, it focuses on the most interesting one. A person is interesting to the robot if the person looks at it or talks. Of course, the robot prefers persons that talk while facing it. The robot focuses a person by turning its camera into the direction of that person. Moreover, an avatar-like face is presented on the robot's display which also looks in the same direction. A user can start an interaction by greeting the robot with, e.g., "Hello BIRON". Then, the robot keeps this user in its focus and can not be distracted by other persons talking. Next, the user can ask the robot to follow him to another place in order to introduce the robot to new objects. While the robot follows a person it tries to maintain a constant distance to the user and informs the person if it moves too fast. When the robot reaches a desired position the user can instruct the robot to stop. Then, the user can ask the robot to learn new objects. In this case the camera focuses the person's torso [10] to get the hands of the person in its field of view. After pointing to a position and giving spoken information like "This is my favorite cup", the object attention system centers the referred object. After the object is recognized, all information acquired is added to the scene model. If the user says "Good-bye" to the robot or simply leaves while the robot is not following the user, the robot assumes that the current interaction is completed and looks around for new potential communication partners. For a detailed evaluation on how users interact with the robot see [22].

Besides these interaction capabilities the robot can also be instructed to autonomously move to a specific location. After reaching the goal the robot tries to find a communication partner again. If a failure occurs along the way, the robot also looks for a communication partner in order to ask for help.

IX. SUMMARY

In this paper we presented an architecture concept for human-robot interaction and its implementation on our robot BIRON. After reviewing methodologies for robot control systems and existing systems incorporating human-robot interaction, details of our agent-based architecture concept which focuses on the integration of human-robot interaction capabilities were presented. As the development of the central part in a robot system that connects deliberative-like control to reactive control is challenging, we paid special attention to the design of the presented execution supervisor. It is based on an augmented finite state machine which is specified in XML and thus is highly generic. An event queue is used to manage all incoming events which are received asynchronously. The data flow is also based on XML in order to satisfy non-functional aspects. Moreover, the execution supervisor is capable of synchronizing events and due to its scalability it can be easily extended to incorporate new functionalities. We explained how the execution supervisor interacts with other components of our system as basis for advanced human-robot interaction in the home tour scenario. Finally, we presented the current interaction capabilities of the overall system which are realized with the proposed architecture.

REFERENCES

[1] R. C. Arkin. Path planning for a vision-based autonomous robot. In *Proc. SPIE Conf. on Mobile Robots*, pages 240–249, Cambridge, MA, 1986.

[2] R. C. Arkin. Motor schema-based mobile robot navigation. *Int. Journal of Robotics Research*, 8(4):92–112, 1989.

[3] R. C. Arkin. *Behavior-Based Robotics*. MIT Press, Cambridge, MA, 1998.

[4] H. Asoh, Y. Motomura, F. Asano, I. Hara, S. Hayamizu, K. Itou, T. Kurita, T. Matsui, N. Vlassis, R. Bunschoten, and B. Kröse. Jijo-2: An office robot that communicates and learns. *IEEE Intelligent Systems*, 16(5):46–55, 2001.

[5] R. Bischoff and V. Graefe. Demonstrating the humanoid robot *HERMES* at an exhibition: A long-term dependability test. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems; Workshop on Robots at Exhibitions*, Lausanne, Switzerland, 2002.

[6] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, 1986.

[7] R. A. Brooks. Elephants don't play chess. *Robotics and Autonomous Systems*, 6(1–2):3–15, 1990.

[8] W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1–2):3–55, 1999.

[9] R. James Firby. *Adaptive Execution in Dynamic Domains*. PhD thesis, Yale University, New Haven, CT, 1989.

[10] J. Fritsch, M. Kleinehagenbrock, S. Lang, G. A. Fink, and G. Sagerer. Audiovisual person tracking with a mobile robot. In F. Groen, N. Amato, A. Bonarini, E. Yoshida, and B. Kröse, editors, *Proc. Int. Conf. on Intelligent Autonomous Systems*, pages 898–906, Amsterdam, 2004. IOS Press.

[11] J. Fritsch, M. Kleinehagenbrock, S. Lang, T. Plötz, G. A. Fink, and G. Sagerer. Multi-modal anchoring for human-robot-interaction. *Robotics and Autonomous Systems, special issue on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, 43(2–3):133–147, 2003.

[12] E. Gat. Integrating planning and reacting in a heterogenous asynchronous architecture for controlling real-world mobile robots. In *Proc. Nat. Conf. on Artificial Intelligence (AAAI)*, pages 809–815. AAAI/MIT Press, San Jose, CA, 1992.

[13] E. Gat. On three-layer architectures. In D. Kortenkamp, R. P. Bonasso, and R. Murphy, editors, *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems*, chapter 8, pages 195–210. MIT Press, Cambridge, MA, 1998.

[14] B. P. Gerkey, R. T. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proc. Int. Conf. on Advanced Robotics*, pages 317–323, Coimbra, Portugal, 2003.

[15] A. Haasch, S. Hohenner, S. Hüwel, M. Kleinehagenbrock, S. Lang, I. Toptsis, G. A. Fink, J. Fritsch, B. Wrede, and G. Sagerer. BIRON – The Bielefeld Robot Companion. In *Proc. Int. Workshop on Advances in Service Robotics*, pages 27–32, Stuttgart, Germany, 2004. Fraunhofer IRB Verlag.

[16] M. Hans and W. Baum. Concept of a hybrid architecture for Care-O-bot. In *Proc. IEEE Int. Workshop on Robot and Human Interactive Communication (ROMAN)*, pages 407–411, Bordeaux/Paris, France, 2001. IEEE.

[17] M. J. Huber. JAM: A BDI-theoretic mobile agent architecture. In *Proc. Int. Conf. on Autonomous Agents (Agents)*, pages 236–243, Seattle, WA, 1999.

[18] F. F. Ingrand and F. Py. An execution control system for autonomous robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 1333–1338, Washington DC, USA, 2002.

[19] G. Kim, W. Chung, M. Kim, and C. Lee. Tripodal schematic design of the control architecture for the service robot PSR. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2792–2797, Taipei, Taiwan, 2003.

[20] B. J. A. Kröse, J. M. Porta, A. J. N. van Breemen, K. Crucq, M. Nuttin, and E. Demeester. Lino, the user-interface robot. In *European Symposium on Ambient Intelligence (EUSAI)*, pages 264–274, Veldhoven, Netherlands, 2003.

[21] S. Lang, M. Kleinehagenbrock, S. Hohenner, J. Fritsch, G. A. Fink, and G. Sagerer. Providing the basis for human-robot-interaction: A multi-modal attention system for a mobile robot. In *Proc. Int. Conf. on Multimodal Interfaces*, pages 28–35. ACM, 2003.

[22] S. Li, M. Kleinehagenbrock, J. Fritsch, B. Wrede, and G. Sagerer. "BIRON, let me show you something": Evaluating the interaction with a robot companion. In *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, Special Session on Human-Robot Interaction*, The Hague, The Netherlands, 2004. to appear.

[23] M. Lindström, A. Orebäck, and H. I. Christensen. BERRA: A research architecture for service robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, volume 4, pages 3278–3283, San Francisco, CA, 2000.

[24] M. J. Matarić. Behavior-based control: Examples from navigation, learning, and group behavior. *Journal of Experimental and Theoretical Artificial Intelligence, special issue on Software Architectures for Physical Agents*, 9(2–3):323–336, 1997.

[25] M. J. Matarić. Learning in behavior-based multi-robot systems: Policies, models, and other agents. *Cognitive Systems Research, special issue on Multi-disciplinary studies of multi-agent learning*, 2(1):81–93, 2001.

[26] M. Montemerlo, J. Pineau, N. Roy, S. Thrun, and V. Verma. Experiences with a mobile robotic guide for the elderly. In *Proc. Nat. Conf. on Artificial Intelligence (AAAI)*, pages 587–592, 2002.

[27] N. J. Nilsson. *Principles of Artificial Intelligence*. Springer Verlag, Berlin, Germany, 1982.

[28] S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Probabilistic algorithms and the interactive museum tour-guide robot Minerva. *International Journal of Robotics Research*, 19(11):972–999, 2000.

[29] N. Tschichold, S. Vestli, and G. Schweitzer. The service robot MOPS: First operating experiences. *Robotics and Autonomous Systems*, 34(2-3):165–173, 2001.

[30] S. Wrede, J. Fritsch, C. Bauckhage, and G. Sagerer. An XML based framework for cognitive vision architectures. In *Proc. Int. Conf. on Pattern Recognition*, Cambridge, UK, 2004. to appear.