A Unified Robotic Software Architecture for Service Robotics and Networks of Smart Sensors

Daniel Westhoff and Jianwei Zhang

Institute Technical Aspects of Multi-modal Systems
Dept. of Informatics, University of Hamburg
Hamburg, Germany

Abstract. This paper proposes a novel architecture for the programming of multi-modal service robots and networked sensors. The presented software framework eases the development of high-level applications for distributed systems. The software architecture is based upon the Roblet-Technology, which is an exceptionally powerful medium in robotics. The possibility to develop, compile and execute an application on one workstation and distribute parts of a program based on the idea of mobile code is pointed out. Since the Roblet-Technology uses Java the development is independent of the operation system. The framework hides the network communication and therefore greatly improves the programming and testing of applications in service robotics. The concept is evaluated in the context of the service robot TASER of the TAMS Institute at the University of Hamburg. This robot consists of a mobile platform with two manipulators equipped with artificial hands. Several multimodal input and output devices for interaction round off the robot. Networked cameras in the working environment of TASER provide additional information to the robot. The integration of these smart sensors shows the extendability of the proposed concept to general distributed systems.

1 Introduction

Robotic systems are becomeing more and more complex. The number of constitutional parts that make up current robotic research platforms is increasing. A multitude of sensors can be found in these robots: tactile sensors from basic bumper switches to force and torque sensors, range measuring systems like infrared, ultra-sonic, radar and laser based sensors or vision systems including cameras as different as low-cost web-cams and high-dynamic-range cameras. On the actuator side one finds mobile robot platforms with a variety of drive systems, walking or climbing robots, robot arms with different degrees of freedom or complex multi-finger robotic hands. In service robotics all these are combined in autonomous mobile manipulators that accomplish tasks in a diversity of applications.

Over the last years, the research community has come to realize that the ambitious objectives of robotic research can only be reached based on solid software architectures. These architectures must support the requirements of the heterogeneous modern robot systems. Briefly summarized, the main requirements are:

hardware abstraction, extendability, scalability, limited run-time overhead, actuator control, modularization, support for networked computing, simplicity, consistency, completeness, support for multiple operating systems. [1] and [2] have conducted surveys and evaluations of existing software systems. They provide a good elaboration on the merits and demerits of these architectures.

In this paper we present our robotic software framework, which is based on Roblet-Technology [3] and thereby meets the above challenges while enabling a programmer to easily develop advanced applications for service robots. First, we discuss related software architectures which have been proposed for robotic applications in the last years. Second, a short introduction to our robot system TASER is given which points out the need for a unifying development platform for robotic systems. Then, the main features of our framework are presented. Most important is the framework's ability to integrate existing solutions to specific robotic problems, making it a component-oriented approach. It is specially designed for networked applications. The system features a layer that encapsulates network programming and hides it from its user. A variety of hardware devices can be integrated into an application using our technique. We give examples how we used the framework to develop complex applications using a variety of robot components. Thereby, a layer of abstraction generalizes access to similar devices. In the conclusion we discuss that developed applications can be transferred to other robotic systems without major changes.

2 Related research

This section gives an overview of existing software architectures for service robots. Recently, a workshop during the 2004 conference on Intelligent Robots and Systems (IROS) tried to list the various research activities in the field of robotic middleware [4]. In the following, some of these activities are discussed. Besides, further related research projects are stated.

In 2004 the *Orca project* [5] emerged from the *OROCOS project* [6] and focused on mobile robot systems. It adopts a component-based software engineering approach using *Ice* for communication and the description of interfaces. The project's goals are to enable and to simplify software reuse and to provide a generic repository of components. The use of different middleware packages for inter-component communication is extensively discussed on the project's home page. Beside writing custom middleware, the use of CORBA and XML-based technologies is compared to Ice. Orca is available for various operating systems like Windows or Linux.

[7] introduces the *Player/Stage project*, a client-server framework to enable research on robot and sensor systems. It provides a network interface to a variety of robot and sensor hardware and to multi-robot simulators. Multiple concurrent client connections to the servers are allowed. Client applications connect over TCP sockets. The project's server software and the simulators are limited to Unix-like operating systems.

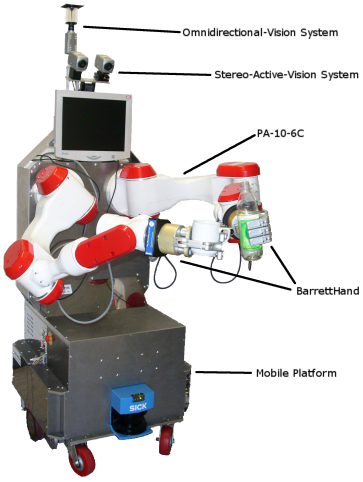

Fig. 1. The service robot TASER with its constitutional parts labeled.

Fig. 2. Generalized motion trajectories within the laboratory of the TAMS institute (top) and a snapshot of the tracking process (bottom).

MARIE [8] is a design tool for mobile and autonomous robot applications. It is mainly implemented in C++ and it uses the *ADAPTIVE Communication Environment (ACE)* for communication and process management.

In 2002 *Evolution Robotics* introduced the commercial *Evolution Robotics Software Platform (ERSP)* for mobile robots [9]. It is a behavior-based, modular and extensible software available for Linux and Windows systems. The main components that are included are vision, navigation and interaction.

The ideas for the software framework explained in this paper emerged from previous work on a mobile robot working in a biotechnological laboratory [10]. An easy-to-use script language was proposed to define high-level work sequences. The scripts are parsed by the robot's control software and the robot fullfils the defined task. This encourages the idea of simplifying the programming of robots but lacks the flexibility of a widespread programming language including network programming for distributed systems.

3 Service robot and hardware set-up

The **TA**MS **Se**rvice **R**obot (TASER) is a mobile robot built from standard components. On top of the mobile basis, there are two manipulators with three-finger robotic hands and force sensors and a man-machine interface including a monitor for visual and loudspeakers for aural feedback. A stereo camera system for active vision and an omni-directional vision system provide visual input. Microphones are used as aural input sensors. Figure 1 shows TASER. The robot operates during normal workdays in our office environment.

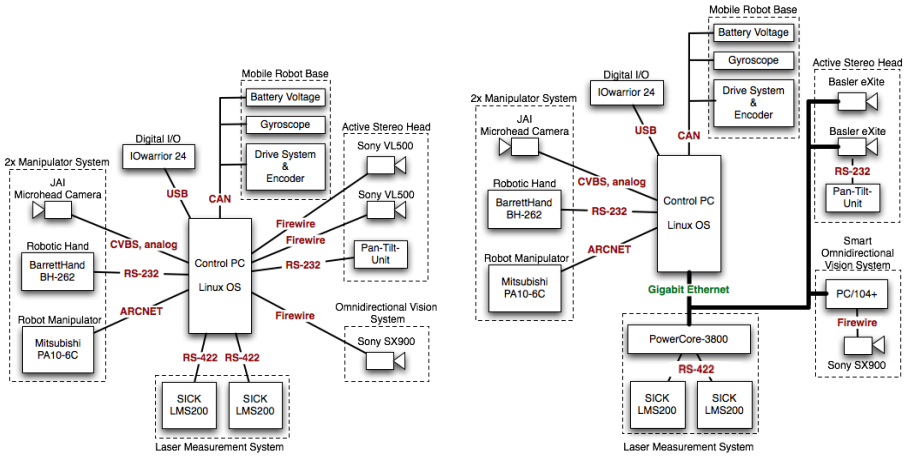

Fig. 3. The hardware set-up of TASER: The initial set-up with various bus systems (left) and the current refined set-up based on smart sensors (right). Both set-ups are connected to the stationary sensors and workstation computers via wireless LAN.

In addition to the on-board sensors of the robot, there are stationary cameras that monitor the environment. The data of the stationary cameras is available to the robot and to other networked applications. An exemplary application using both the robot's sensors and the stationary vision systems is a people tracking software [11]. This software tracks people, learns and generalizes motion trajectories of these people and provides predictions of trajectories to other applications. The service robot uses these predictions within its path-planning module to intercept people for interaction. Figure 2 shows the generalized trajectories for a part of the robot's working environment besides a snapshot of the camera tracking.

One fundamental idea we followed during the development of TASER was the integration of smart sensors and devices. Smart sensors are sensor devices that include additional computing power, which can be used for the pre-processing of sensor measurements before the data is made available to other system components. Pre-processing can be data compression, filtering or rudimental feature extraction. Smart sensors, actuators and control computers are connected via Ethernet. This has enabled us to unite and reduce bus systems used on the robot. Figure 3 points this out. Additionally, one-to-many connections can be implemented. For example, the laser range finders of the robot are used in parallel by the robot's low-level collision and obstacle avoidance system, the self-localization algorithm, both running on the control computer of the robot, and by the people tracking application running on a workstation computer in the laboratory [12].

These exemplary applications are strongly based on Roblets to combine the involved components. The following section explains the idea of Roblets and our robotic development environment.

4 Software design and architecture

All higher-level applications controlling the robot are programmed using the Roblet-Framework [3]. The basics of the proposed framework are realized with Java. Roblet-Technology is a client-server architecture where clients can send parts of themselves, referred to as Roblets, to a server. This idea of mobile code is well-known since the 1990s and mostly applied in software agents. With the Roblet-Framework, mobile code was introduced as a key feature in robotics software. Additionally, the use of technologies which are provided by Java's standard libraries results in stable and reliable applications, since the same functionality is used by millions of developers and users on the Internet.

Each sensor of a robotic system is encapsulated by a Roblet-Server providing interfaces to access sensor measurements. In the same way, actuators are encapsulated and provide basic interfaces on an entity level. For example, for the mobile platform this means commands like move forward or turn to some specific orientation can be executed.

Servers mostly wrap already existing lower-level software like the real-time control subsystems for the manipulators. Besides hardware encapsulation, existing software packages for robotic problems are available via Roblet-Servers. The path planning component of [10] is one such package and an example for the component-based reuse of software within our framework.

A basic Roblet-Server and a library for client applications are commercially available and supported by the *genRob GmbH*. It is freely available for research projects and extendable by self-programmed modules.

Modules are loaded when the Roblet-Server is started. These are meant to encapsulate a class of similar functionality. For the robot TASER we developed several modules. Figure 4 gives an overview of the current state of our system's implementation.

Client applications use units to query functionality. Units are provided and implemented by modules. They are specified as Java interfaces. In this way, different modules can implement the same units and therefore allow the use of similar hardware in a unified way. To point this out, a camera application is given as an example of a distributed application in figure 4. It uses a unit *camera* which is implemented by different modules. A directory service informs the client application about Roblet-Servers that are added or removed from the overall system during runtime.

Java classes, which implement the Roblet interface can be sent to a server from a client program and are executed on the remote side. Similar to Java applets this technique allows the sending of code over a network. Now, this enables the developer of a client application to decide were he wants the code to be run during runtime. Communication channels between servers and client applications can be created as needed by the Roblets. This leads to a novel flexibility for load distribution concerning computing power as well as the use of network bandwidth. Additionally, only one programming language is needed to develop higher-level applications.

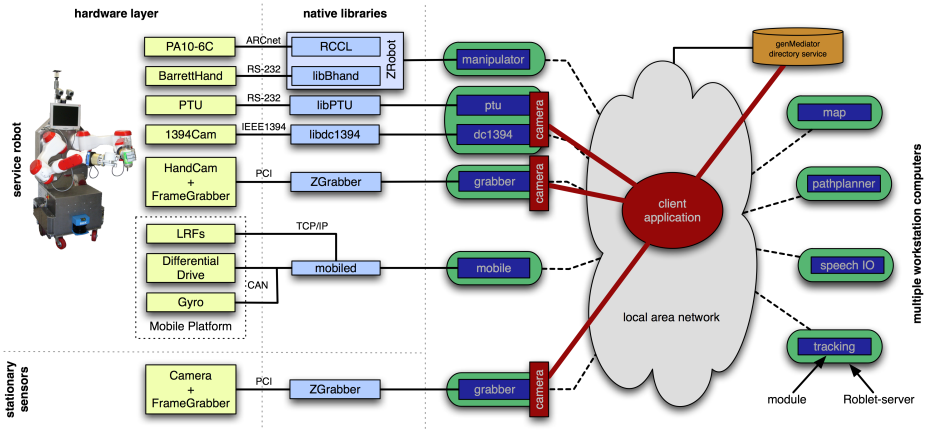

Fig. 4. The software architecture of the robot TASER. A client application uses the same unit *camera* implemented by the two different modules *grabber* and *dc1394* on three servers to display images. These images from the robot's cameras and the stationary cameras are transmitted and displayed synchronously.

In the camera example we execute compression and streaming algorithms within a Roblet before transmitting an image from a server to the client. This saves bandwith on the wireless network. During runtime the Roblet determines the quality of the wireless connection and adopts the compression level.

In the same way secure communication channels can be established. For a wireless connection the Roblet could decide to use stronger encryption algorithms if sensitive data is to be transmitted.

5 Conclusion

The presented software architecture enables the building of high-level applications for service robots using standard components for robots as well as specialized hard- or software. The software architechture of the robot based on the Roblet-Technologie is a powerful medium for robots. The feature of running client programs as a distributed software offers the possibility to run algorithms which need great computation power on different machines which provide this power. The type of communication, e.g. encrypted or compressed communication, can be changed during runtime. Each client application can use its individual and appropriate type of communication.

The next step will be to implement further software to improve the usability of the robot system and create a toolbox of reusable program parts. In this step the variety of the high-level functions like object grasping and multimodal interaction will be increased. Furthermore, the possibilities of autonomous navigation and map building will be extended.

6 Bibliography

1. A. Orebäck and H.I. Christensen: *Evaluation of Architectures for Mobile Robotics*. Autonomous Robots, vol.14(1), pp. 33-49, Springer, Netherlands, 2003.
2. J. Kramer and M. Scheutz: *Development Environments for Autonomous Mobile Robots: A Survey*. Autonomous Robots, vol. 22(2), pp. 101-132, Springer, Netherlands, 2007.
3. D. Westhoff, H. Stanek, T. Scherer, J. Zhang and A. Knoll: *A flexible framework for task-oriented programming of service robots*. Robotik 2004, VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, VDI-BErichte, Germany, 2004.
4. *Workshop on Robot Middleware towards Standards*, Int. Conf. on Intelligent Robots and System (IROS'04), Sendai, Japan, 2004, `http://www.is.aist.go.jp/rt/events/20040928IROS.html`.
5. H. Bruyninckx: *Open robot control software: the OROCOS project*, Proc. of the 2001 IEEE Int. Conf. on Robotics and Automation (ICRA'01), volume 3, pages 2523–2528, Seoul, Korea, 2001.
6. A. Brooks, T. Kaupp, A. Makarenko, A. Orebäck, S. Williams: *Towards Comonent-Based Robotics*, Proc. of the 2005 IEEE/RSJ Int. Conf. on Intellegent Robots and Systems (IROS'05), Alberta, Canada, 2005.
7. B.P. Gerkey, R.T. Vaughn, K. Stoy, A. Howard, G.S. Sukhatme, M.J. Mataric: *Most Valuable Player: A Robot Device Server for Distributed Control*, Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'01), pages 1226–1231, Wailea, Hawaii, 2001.
8. C. Cote, D. Letourneau, F. Michaud, J.-M. Valin, Y. Brousseau, C. Raievsky, M. Lemay, V. Tran: *Code Reusability Tools for Programming Mobile Robots*, Proc. of the 2004 IEEE/RSJ Int. Conf. on Intellige Robots and Systems (IROS'04), pages 1820–1825, Senda, Japan, 2004.
9. N. Karlsson, M.E. Munich, L. Goncalves, J. Ostrowski, E. Di Bernado, P. Pirjanian: *Core Tehnologies for service Robotics*, Proc. of the 2004 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'04), Senda, Japan, 2004.
10. T. Scherer: *A mobile service robot for automisation of sample taking and sample management in a biotechnological pilot laboratory*, University of Bielefeld, Ph.D Thesis, 2005, `http://bieson.ub.uni-bielefeld.de/volltexte/2005/775/`.
11. M. Weser, D. Westhoff, M. Hüser and J. Zhang: *Real-Time Fusion of Multimodal Tracking Data and Generalization of Motion Patterns for Trajectory Prediction*. Proc. of the 2006 IEEE Int. Conf. on Information Aquisition (ICIA'06), China, 2006.
12. H. Bistry, S. Phölsen, D. Westhoff, J. Zhang: *Development of a smart Laser Range Finder for an Autonomous Service Robot*. Proc. of the 2007 IEEE Int. Conf. on Integration Technology (ICIT'07), China, 2007.