



14TH INTERNATIONAL CONFERENCE ON ADVANCED ROBOTICS

Munich, Germany
June 22nd to 26th, 2009

<http://www.icar2009.org>

Sponsored by the German Robotics Society (DGR)
Technically Co-sponsored by IEEE Robotics and Automation Society (RAS)

ICAR 2009 Workshop **June 23, 2009 – Munich, Germany**

Title

Rapid Application Development in Robotics: On the role of re-use and adaptation of system components, middleware, and control architectures

Abstract

The concept of "reusability" and "configurability" of software components is becoming more and more important in the development of modern robotic systems. These much-neglected nonfunctional requirements are key enablers of the innovative, multi-vendor, high added value robotics research, industrial manufacturing and professional services.

Reuse and related concepts are the main driving force in defining the ability of a software system to respond to demands of changing robotics trends. RoSta Middleware and Architecture activities performed exhaustive survey and comparison of the existing state of the art in robot software. The results of the survey confirm that most of the approaches are "from the scratch" systems without extra notable progress in research. One of the main reasons was that most of the previously produced results (in the form of software and experimental data) were barely interchangeable (inconsistent syntax/semantics) or reusable/inter-operable (no common software models for communication, components etc). One of the main objectives of this workshop is to address this issue of "reusable/interoperable/ interchangeable" software by defining initial guidelines/decisions and identifying trade-off points and future research projects.

Program

9:00 -- 9:05	Welcome
9:05 -- 9:30	The Use of Reuse: A White Paper (<i>Erwin Prassler and Azamat Shakhimardanov</i>)

ICAR 2009 Workshop, June 22, 2009 – Munich, Germany

Faster Application Development in Robotics: On the role of re-use and adaptation of system components, middleware, and control architectures in shortening the development cycle

09:30 - 10:00	A reusable API for sensor-based robot motion specification and control (<i>Herman Bruyninckx</i>)
10:00 - 10:30	Reuse of robot software in GearBox, Player and OpenRTM (<i>Geoffrey Biggs</i>)
10:30 - 11:00	Coffee break
11:00 - 11:30	Building middleware-independent robotics software components (<i>Anthony Mallet</i>)
11:30 - 12:00	Principles for layered software in robot control architectures and middleware (<i>Klas Nilsson</i>)
12:00 - 12:30	Techniques to build and reuse software artifacts (<i>Davide Brugali</i>)
12:30 - 13:00	BRICS - Best Practise in Robotics (<i>Rainer Bischoff</i>)
13:00 - 14:00	Lunch
14:00 - 15:25	Discussion and Recommendations
15:30 - 15:55	Coffee break
16:00 - 16:15	Discussion and closing

Work shop organizers: Erwin Prassler, Hochschule Bonn-Rhein-Sieg, Germany
Azamat Shakhimardanov, Hochschule Bonn-Rhein-Sieg, Germany
Klas Nilsson, University of Lung, Sweden

Invited Speakers: Geoffrey Biggs, AIST, Japan
Anthony Mallet, LAAS CNRS, France
Herman Bruyninckx, Katholieke Universiteit Leuven, Belgium
Davide Brugali, University of Bergamo

ICAR 2009 Workshop, June 22, 2009 – Munich, Germany

Faster Application Development in Robotics: On the role of re-use and adaptation of system components, middleware, and control architectures in shortening the development cycle

Geoffrey Biggs**Title:** Reuse of robot software in GearBox, Player and OpenRTM.**Abstract:**

This talk will discuss experiences with the open-source GearBox project and the Player and OpenRTM robot software architectures. These projects each take a different, and in some ways complimentary, approach to reuse of robot software. The benefits of splitting reuse across multiple levels will be discussed, as well as the differences in reuse experienced between the client/server-based Player and the component-based OpenRTM. The speaker's experiences with starting a new open-source project for robotics with an explicit primary goal of high reuse will also be discussed, illustrating some of the pitfalls that can be encountered in such an endeavour.

Anthony Mallet**Title:** Building middleware-independent robotics software components.**Abstract**

Using component-based software architectures for robotics has nowadays reached a great consensus. Software components endow robotics systems with a great amount of interesting properties: components are usually not application specific, independent of other components, encapsulate internal details of algorithms and provide a clean interface. For complex systems, component are also probably the only viable solution to the software reuse problem. Beyond this consensus, roboticists are still faced to the crucial problem of what could be called "the middleware choice". The middleware is an important part of components architecture and has a major influence on the components design. Furthermore, the selection of this middleware has to be done early during the design process of a robotic application. An error at this step can have dramatic consequences, mostly because software components rely a lot on the middleware primitives. This problem is especially preeminent in the context of research projects, where several teams have to share software that was not originally developed from the same base choices. Although a lot of efforts has been put in the standardization (CORBA, RT-Middleware to name just a few), middlewares are usually not compatible between each other. In this presentation, I will describe the internal project that is conducted in the context of the "robotic software platform" of the LAAS/CNRS to tackle this problem. The main idea is to be able to make the software components fully independent of the actual middleware that they will use. This is done by separating the algorithmic software from the core of the component and providing a formal description of the internals of the component. Generic, interchangeable templates specific to the middleware can then be instantiated and compiled to form the final component. All this is a work in progress that started recently but preliminary result will still be presented. A new version of the open-source GenoM component generator tool, that has been developed and used at LAAS for the last 15 years, will be available in the short-term and will integrate these ideas. To conclude, I will also quickly mention a software packaging system that was developed recently at LAAS/CNRS. This project is called "robotpkg" and is geared toward simplifying the compilation and installation of robotic software by using the concept of "source software packages" that can be found on any modern Unix distribution.

Herman Bruyninckx**Title:** "A reusable API for sensor-based robot motion specification and control"**Abstract:****ICAR 2009 Workshop, June 22, 2009 – Munich, Germany**

Faster Application Development in Robotics: On the role of re-use and adaptation of system components, middleware, and control architectures in shortening the development cycle

Motion specification and control are quite mature fields. Not in the least for the traditional position-controlled industrial robot arms, but also for the still more research-oriented redundant kinematic chains such as mobile manipulators or humanoid robots. Nevertheless, no (commercial nor opensource) SDKs (or API standards) exist at this moment, and every manufacturer or research institute is designing and implementing its own version. Major reasons for this extreme fragmentation are: slightly different use-cases (e.g. different sensors or kinematic chains), different numerical requirements (e.g., real-time control or motion planning), and a lack insights in how and why to design generic APIs, and standardizing them without removing the flexibility for reuse in all possible use case. The presented API has an answer to all above-mentioned problems, and is presented to the community for critical discussion and possible refactoring.

Klas Nilsson:

Title: Principles for layered software in robot control architectures and middleware.

Abstract: A variety of software a system engineering principles are being used within the robot community. For instance, partly due to due to similarities in resource constraints, embedded systems design is one of these source of input. A common principle in both software architectures and in protocol stacks is layering, where standardized APIs to each layer promote reuse, portability, and interoperability. On top of this, standardized reference architectures (such as AUTOSAR for vehicles) is intended to further improves the situation by standardizing middleware and component interfaces/types. That is clearly an inspiration for the robotics community that struggle with lack of reuse and too expensive system developments. However, a more detailed examination reveals that due to rapidly changing technologies, heterogeneous hardware and requirements, wide variability in competences and preferences, and so on, the standardization approach does not quite work in robotics (with the exception of safety where there are legal/liability reasons); standardization is in many cases too slow a process to keep up with the needs for changeability. The question then is: Can we still find some means (key architectural abstractions and mechanisms) such that reuse is promoted in an evolving world? The answer "yes" is given with a number of suggestion how to proceed, including some new perspectives of layered software.

Davide Brugali:

Title: Techniques to build and reuse software artifacts

Abstract: Open

ICAR 2009 Workshop, June 22, 2009 – Munich, Germany

Faster Application Development in Robotics: On the role of re-use and adaptation of system components, middleware, and control architectures in shortening the development cycle