Using Microsoft® Robotics Studio for the design of generic robotics controllers: the robuBOX® software

Damien Sallé, Marc Traonmilin, Joseph Canou and Vincent Dupourqué.

***Abstract*—This paper introduces the robuBOX®, being developed by ROBOSOFT and based on Microsoft® Robotics Studio. The ROBOSOFT robuBOX® is a software for the design of generic robotics controllers implementing advanced robotics functions and behaviors. Software for robots is very different than software for computers: a robot is more than a computer with wheels and motors. This statement led ROBOSOFT to define new objects, called robuter®.**

The robuBOX® provides integrators and manufacturers with an off-the-shelf set of software and hardware modules embedding 80% of the complexity of intelligent service robots. It is based on reference designs for the robots and control architectures that allows to plug, configure and play a robuBOX® on a machine to make it a professional service robot. But it also provides all the required openness for researchers to design a specific control architecture.

The robuBOX® has already been implemented in many service robotics applications that are shortly described in this paper. The benefits of such software is then evaluated with respect to existing solutions, in the context of an industrial company, European leader in robotics for professional services.

I. Introduction

THE 20th century has opened the area of service robotics. The ageing population leads to less workers and more demanding services to people. Service robots help human beings in their everyday life for basic services such as transport, cleanliness, security, care…

The 21st is seeing the birth of industrial-grade service robots. There is an emerging but promising market for applications such as automatic shuttles for transport of people in city centers, hospitals, airports, theme parks…, automatic floor and window cleaning in vast or risky areas, robotic assistance to elderly and disabled, etc.

In developing these very specific robotic solutions since 1985, ROBOSOFT discovered that software for robots was very different than software for computers. To make a long story short, we could say that there are two major differences:

The first is Time Management. With a regular computer, managing time generally means going as fast as possible. All technical improvements, for both hardware and algorithms, aim at going faster or processing more in the same amount of time. For robots, because they are physical devices, the laws of physics apply. A robot has to follow a path, which means that its position depends on time. It also has to take into account mass, inertia and other forces. In other words, commands must be computed at a given and regular time, rather than as fast as possible.

The second difference between software for robots and software for computers involves robustness. Everybody knows the « blue screen syndrome ». When computer software fails for any reason you have to reboot, the worst consequence being that you have a few minutes you can use to drink a cup of coffee and you may have lost a few minutes of unsaved work. When the software of a robot fails, the robot may stop -- and this is the “best case” scenario! Imagine that rather than stopping, it would continue or accelerate and go crazy, hitting walls or objects and injuring people! Robotics software must thus involve many more safeguards to prevent these kinds of “horror stories”.

Specialists know how far these two simple considerations can make things difficult! We had to abandon our notion that a robot was simply a computer with wheels and motors. Developing software for robots led to something else which ROBOSOFT calls a robotized-computer, or a “Robuter®”!

In order to allow fast and esay robotization these robuters as well as machines for professional services, ROBOSOFT has developed the robuBOX concept. It integrates 80% of the complexity of intelligent service robots, yet making possible the production of service robots at industrial cost and quality. To provide robotics integrators with an open and flexible programming environment, the robuBOX is based on the Microsoft Robotics Studio.

II. State of the art in components based control architecture software for robotics

The robuBOX is ROBOSOFT’s 3rd generation of robotic controller software since its creation in 1985. The first generation, Albatros [1] was a pure internal development with a real time OS and some generic functions to perform low to medium level control of the platforms.

Then the development of complex robotized vehicles drove requirements for distributed architectures on numerous microcontrollers and PCs. A joint development with INRIA,

D. Sallé is with the French company Robosoft (phone: +33-5-59-41-53-60; fax: +33-5-59-41-53-79; e-mail: damien.salle@robosoft.fr).

M. Traonmilin is with the French company Robosoft (phone: +33-5-59-41-53-60; fax: +33-5-59-41-53-79; e-mail: marc.traonmillin@robosoft.fr).

J. Canou is with the French company Robosoft (phone: +33-5-59-41-53-60; fax: +33-5-59-41-53-79; e-mail: joseph.canou@robosoft.fr).

V. Dupourqué is with the French company Robosoft (phone: +33-5-59-41-53-60; fax: +33-5-59-41-53-79;e-mail: vincent.dupourque@robosoft.fr).

the French Institute for Computer Science and Automation, led to the development of our second generation of controller software: iCORE, based on Syndex software development methodology [2].

In the past months, the involvement of a new category of clients has made a strong growth in the robotics for professional services market. These companies have limited competencies in robotics. So the market requirements are shifting from robotic technology to fully operational and cost-effective robotized solution for service applications.

To fully answer these new requirements, ROBOSOFT had to move to a 3rd generation of robotic control system. A broad state of the art and analysis of the available solutions, both open-source and proprietary, have been carried out with the following conclusion: there is not a single software that can be considered as a reference in the field and that would ensure efficient improvement and maintenance in the medium/long term.

Another point is that almost every research lab involved in robotics engineering is developing its own environment and tools for the development of their robots software. Most of them are then using real time operating systems such as VxWorks or Linux RTAI, on top of which they build their own kernel modules and applications.

For research labs more involved in the artificial intelligence fields, a general trend is the use of existing middleware for robotics in order to re-use this existing robotics competency.

These middleware are a growing research field and are the focus of some dedicated workshops [3]. They can be again differentiated with respect to real-time management: the MCA middleware [4] is a component based architecture software entirely based on RTAI.

On the other hand, the category of non real-time middleware, based on either Linux or Windows, is more popular and more populated. We can cite for example OROCOS [5], ORCA [6], MIRO [7], Player/stage [8], SMARTSOFT [9] and MARIE [10]. All these middleware consider component based or service based approaches to handle the asynchronicity of real environments. They have however their own advantages and pitfalls. One should note the MARIE tentative to merge all these approaches in a single solution that benefits from the advantages of each.

Some of these non real-time middleware have also been used commercially by some robotics companies.

However most of these architectures have the drawbacks of their benefits: they are usually developed and maintained by communities of developers, and most of the time inside a national or European funded project. Indeed, when the project reaches its end because of the graduation of the PhD students or the end of the funding, then these software are more difficult to maintain, more difficult to improve and to adapt to the newest releases of libraries and versions of the OS.

This leads to a fairly high risk for a commercial company when basing its developments on such software. Maintaining and continuously improving industrial quality software products requires time, stability and consistency.

In the mean time Microsoft has announced the development of the Microsoft Robotics Studio (MSRS). As being introduced early to this product development, ROBOSOFT has been able to evaluate the functionalities provided by the MSRS with respect to the existing middleware. The core concept of loosely-coupled service-based control architectures is similar, but the MSRS provides additional very useful functionalities.

These are some of the reasons for ROBOSOFT to choose to support the Microsoft Robotic Studio, as an early adopter.

III. THE MICROSOFT ROBOTICS STUDIO

The Microsoft Robotics Studio is a Windows®-based development environment for academic and commercial developers to easily create robotic applications across a wide variety of platforms. The Microsoft Robotics Studio environment includes:

- An End-to-End Robotics Development Platform, including a visual programming tool, making it easy to create and debug robot applications.
- A lightweight services-oriented runtime, using a .NET-based concurrency library, making asynchronous application development simple.
- A scalable and extensible platform, allowing third-parties to extend the platform by providing additional libraries and services.

Using the Microsoft Robotics Studio, the ROBOSOFT robuBOX core greatly eases and speeds the integration and customization of industrial-grade service robots, letting integrators on their value by working on business requirements.

IV. ROBUBOX: seamless building of industrial-grade service robots

ROBOSOFT provides the robuBOX, a generic and advanced robotics controller software, developed with Microsoft® Robotics Studio and implementing high-level mobile robotic functions like path generation and following, obstacle avoidance, localization and navigation... It aims at quickly robotizing any type of mobile platforms and vehicles.

The robuBOX main target is to provide integrators and manufacturers with an off-the-shelf solution to quickly and easily build standalone or fleets of service robots, such as AGV (Automatic Guided Vehicles), scrubbing machines,

golf cars, and so on. Based on reference designs for both the hardware platforms and control software, it becomes really easy and fast to transform any machine into a professional service robot.

On the other hand, the ROBOSOFT's internal development tools are also made available to highly competent robotics professionals and researchers to allow them to develop any control architecture, using provided hardware and algorithm services, or using their own developments.

So the robuBOX software is made of different solutions, depending on the level of qualification of its user: from highly integrated for specific industrial applications (robuBOX configurator) to fully open for researchers (robuBOX designer).

This set of software solutions, described in the following sub-sections, includes the robuBOX services architecture templates and generic interfaces, as well as efficient robotic algorithms services, implemented for real operations, and sorted by applications, as illustrated in section V.

Finally, the robuBOX enhances the realistic simulation environment provided by the MSRS, as it provides easy integration of existing robots or machines, and last but not least, the capability to use exactly the same control architecture for the simulated robot than for the real robot. This allows of course, a drastic gain in time and development effort for the prototyping and implementation of robotic control architectures.

A. RobuBOX services architecture and generic interfaces

The robuBOX is based on the Microsoft Robotics Studio. It thus uses all the mechanisms available for communication and synchronization between the services.

RobuBOX however greatly simplifies the controller design as it enhances the MSRS intrinsic capabilities for re-use of services, generic interfaces, fast development and prototyping of robotic control architectures. This is achieved through numerous functionalities:

- Definition of generic interfaces between the services: ROBOSOFT, using its 20 years of experience as an advanced service robotics designer and manufacturer, has proposed definitions for standardized interfaces between robotic components and algorithms: how is defined a localization data, a laser scan data, a trajectory data, a differential or a car-like drive, a map of the environment, a landmark extractor etc… They correspond to standardized abstract contracts in the MSRS vocabulary.

- Enhanced dynamic interfacing capabilities: using these generic interfaces, ROBOSOFT has developed software to be able to dynamically replace a running control component by another of the same type.

- Enhanced security and errors management: the robuBOX® includes 3 levels of management of the services:
  - a Node manager that launches the required services on each of the DSS nodes of the possibly distributed application, using a safe-launch procedure,
  - a Device manager that ensures all-time coherence and integrity of the services architecture through heart-beat listening, thus improving the reliability and safety of the application,
  - an Application manager that includes the 2 previous managers and is used to deal with application level services, providing alarms and application integrity status.

- Enhanced debugging and monitoring capabilities: some powerful debugging features are already included in the MSRS such as the capability to access each service status in a web browser, an integrated console for logging info distributed in the application, and of course the debugging environment of Microsoft Visual Studio. However, our experience has shown that these features needed to be enhanced in order to perform really efficient and rapid development of robotic applications.
  The robuBOX thus integrates, within the Application Manager, a generic HMI, collecting all the running services and providing direct access to the data, through graphical visualization: 2D laser scans, trajectory plotting, detected landmarks, generated metric map etc…

The robuBOX architecture and standardized interfaces thus allow to generate powerful services architectures for robotic systems. For example, figure 1 illustrates the robuBOX services architecture for a mobile robot that needs to repeat pre-learnt trajectories.

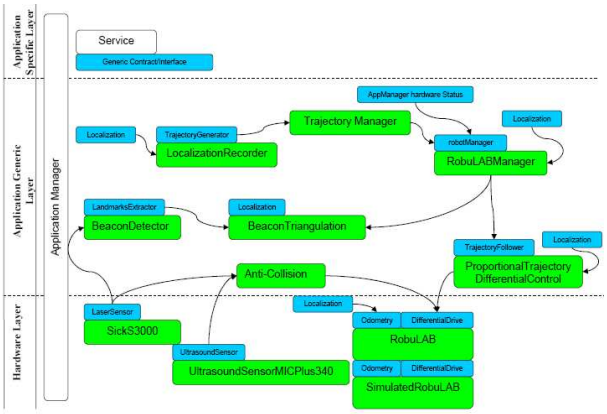

Fig. 1. RobuBOX services architecture example for a trajectory following mobile robot.

Using most of the services of this architecture and adding just a few additional algorithms services, one can very rapidly define a robot performing an explorer task while constructing is own metric map using a SLAM algorithm, as illustrated in figure 2.

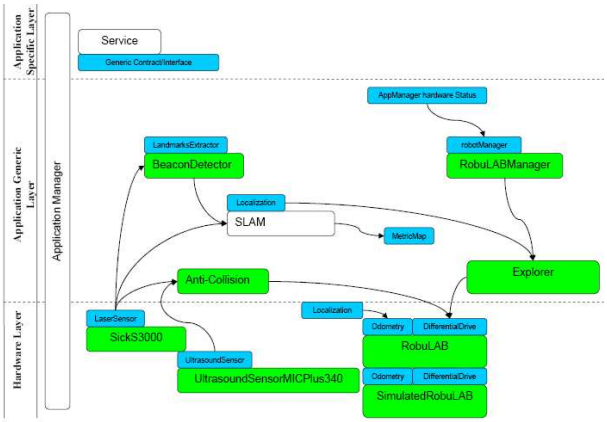

Fig. 2. RobuBOX services architecture example for an explorer robot creating a metric map of its environment.

To further improve the comfort and easiness of development of the robotized application control system, ROBOSOFT has developed an additional software component: the robuBOX designer.

B. RobuBOX designer

The robuBOX designer is a software component that allows to easily generate new robotic architectures from existing services, provided with the application robuBOX packages or developed by third parties.

To generate such architecture, one must first define the hardware components used by the robot, from a non-exhaustive list of supported sensors and drives. Using the generic interfaces defined for these hardware services, the robuBOX designer identifies the existing services that can be used, and provides this list to the user. Using simple drag'n'drop operations, one can thus create the desired architecture, as illustrated in figure 3.

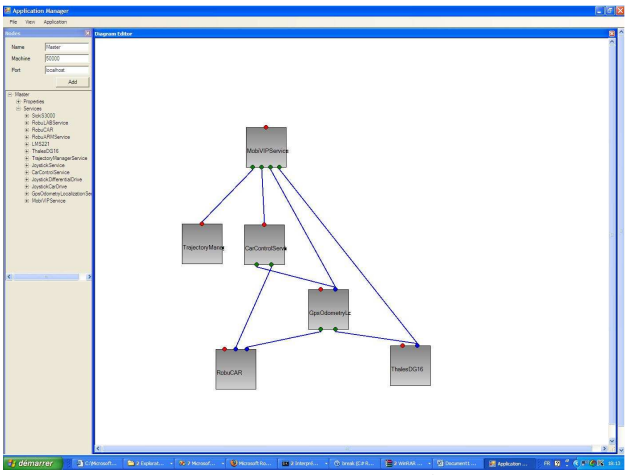

Fig. 3. RobuBOX services architecture example for an explorer robot creating a metric map of its environment.

Again, using the generic interfaces defined by ROBOSOFT, the robuBOX designer validates the communication channels drawn by the user: for example a LaserSensor service cannot be used as direct input for a TrajectoryFollower service.

Each service can be associated to given DSS node for its deployment and execution in a distributed architecture. Each service box can also be configured for constants and parameters using a simple right-clic. This architecture can of course be saved and re-used.

Previous to the deployment, the architecture can be tested and evaluated in the simulation environment provided by MSRS and enhanced by robuBOX, as illustrated in section IV.C.

C. RobuBOX rapid prototyping through realistic simulations

The Microsoft Robotics Studio embeds in its core a physical engine used to perform dynamic simulations of robots. The robuBOX extends its capabilities as it allows to use the exact same control law: the hardware components service are automatically replaced by the simulation services exhibiting the related standardized interfaces. In the example shown in figure 4, both the platforms, the laser range finder and the ultrasound sensors are simulated, but the same architecture as figure 2 is used.

Fig. 4. Example of multi-robots realistic simulation using the robuBOX.

The extension of the MSRS simulation is made through an appropriate use of the entities and the generic contracts defined by ROBOSOFT. This enhancement of the simulations allows to very simply simulate the behavior of the robot. Once debugged and triggered, the exact same architecture is implemented on the real robot.

Of course, this remains a simulation, with all the known limitations and requirements to correctly set the environment physical properties. But our experience up to now and using default parameters is positive: a trajectory following algorithm for a robuCAB (automatic transportation of 4 persons) has shown very similar results in simulation and in real experiments.

D. *RobuBOX configurator*

Another feature provided by the robuBOX is the robuBOX configurator. This software component is used by end-users or integrators of the robuBOX with professional machines, in order to robotize them. Indeed, such targeted audience does not have the required expertise in robotics to define and trigger its own architecture.

The 20 years know-how of ROBOSOFT is provided through the robuBOX application packages: to a given area corresponds a given architecture and a given set of algorithms and services.

The user or integrator, through the robuBOX configurator, thus only has access to configuration of such applicative robuBOX: the type of machine (differential or car-like), the location of the laser sensors on the vehicles, etc…

This one step configuration is the only action an end-user needs to define in order to deploy the robuBOX in its machine and robotize it.

V. THE ROBUBOX AT WORK

The robuBOX products range defines reference designs and reference applications for the integration of the robuBOX in all the targeted applications.

The applications described below have all been implemented in real conditions and environment. Further details for these applications will appear in future conference papers or white papers.

A. *robuBOX-AGV:*

Specifically designed for various kinds of Automated Guided Vehicles, includes all functions performed by AGV : path teaching and repeating, obstacle avoidance, fleet management…

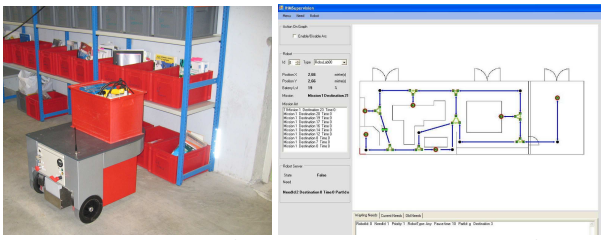

Fig. 5. RobuBOX-AGV for the automatic transportation of goods, used within a fleet of mobile robots.

B. *robuBOX-Cybercar:*

Allows to design systems from single dual mode Cybercars to complex fleet of automatic vehicles, in shuttle or taxi modes. Provides both the Cybercars embedded controllers and the remote fleet management controller.

Fig. 6. RobuBOX-Cybercar used during a 4weeks experimentation in real conditions for the MOBIVIP, French PREDIT funded project.

C. *robuBOX-outdoor:*

For sophisticated all-terrain robots for exploration or surveillance in both autonomous and/or remote control modes.

Fig.7. RobuBOX-Outdoor implemented on a robuROC6.

D. *robuBOX-manipulator:*

Includes all functions for multi-axis serial manipulators and devices, including kinematic model, path generation and redundant kinematic control.

Fig. 8. RobuBOX-Manipulator implemented on a robuARM-S6.10.

VI. EVALUATION OF THE ROBUBOX

The robuBOX benefits, with respect to previously available professional service robotics controller software, are detailed above:

- Efficiency: high level of services allowing developers to save a considerable amount of time and reduce the risk of such complex developments,
- Performance: all services are profiled and optimized for their functions,
- Robustness: the algorithms used in the robuBOX software have been exhaustively used on a variety of robotics platforms over years,
- Appropriate hardware reference design: each robuBOX product comes with an appropriate and

documented hardware reference design as a model for a quick and easy proprietary design,

- Openness: based on Microsoft Robotics Studio®, the robuBOX software is open to current and future MSRS components and third party developments.

However as the robuBOX and MSRS are still under intensive developments, some drawbacks persist. The main potential concern considering the robuBOX is the fact that the MSRS is running on .net and Windows XPembedded. As a consequence, real time behaviors cannot be guaranteed within the services architecture.

So, for the professional service applications targeted by ROBOSOFT, the low level control loops should not be performed directly within the MSRS framework. The robuBOX is thus providing such functionalities using 2 different solutions: for simple robots, intelligent motor controllers connected using CANOpen are performing the PID control loops and synchronization. For more complex robots requiring more tightly coupled control algorithms, ROBOSOFT is using proprietary Syndex-based microcontrollers' boards. In the mean time, ROBOSOFT is investigating developments on Windows CE 6.0 for the next generation of our microcontroller's boards.

VII. Conclusion and Future Work

ROBOSOFT robuBOX® is the ideal way to quickly bring robotic functions and behaviors to any type of new robot or existing vehicles, and build standalone or fleets of service robots, such as mobile platforms, electric wheelchairs, forklifts, containers carriers, vacuum cleaners, scrubbing machines, golf cars, buses, wheel loaders, compactors…

Future work will consist in further increase the number of generic and application specific services available with the robuBOX®, as well as integrating as much real time capabilities as possible as soon as Microsoft Robotic Studio will be released for Windows CE.

References

[1] P. Pomiers, A. Semerano, V. Dupourque and M. Ghrissi, "Trajectory tracking strategies in service robotics applications", Proceedings of the 5th International Workshop on Advanced Motion Control (AMC) 1998, Coimbra, pp 328-333

[2] P. Pomiers and A. Semerano, "SynDEx for Real-Time Applications Implementations", Proceedings of the 4th International Conference on Climbing and Walking Robots CLAWAR, Karlsruhe, Germany, September 2001.

[3] "Robot middleware and Integration Frameworks", B-IT Tutorial, Bonn, Dec 2005.

[4] K.-U Scholl, J. Albiez and B. Gassmann, "MCA – An Expandable Modular Controller Architecture", Proceedings of the 3rd Real-Time Linux Workshop, 2001, Milano, Italy.

[5] www.orocos.org

[6] http://orca-robotics.sf.net

[7] http://smart.informatik.uni-ulm.de/MIRO/

[8] http://playerstage.sourceforge.net/

[9] http://www.rz.fh-ulm.de/~cschlege/orocos

[10] http://marie.sourceforge.net/