Modular Distributed Architecture for Robotics Embedded Systems

Pierre Pomiers, Vincent Dupourqué
Robosoft, Technopole d'Izarbel, 64210 Bidart, France
Tel: +33 5 59 41 53 66
Fax: +33 5 59 41 53 79
E-mail: pierre@robosoft.fr

Abstract

Tomorrow, advanced transportation robotic applications have to be able to cope with various situations and perform many different tasks in a dynamic and changing environment. Furthermore, finding concrete applications in a wide range of user oriented industrial products, such systems, embedding several computing units, have to cope with an increasing demand of interactivity and to support number of non-critical pieces of hardware and software. This whole set of different capabilities needs to be performed reliably and safely over long time periods. To this aim not only advanced programming techniques, but also appropriate control architectures are required. For these reasons, Robosoft proposes both a set of hardware and software components developed from our own experience in the field of automatic transportation of people and goods, which can be easily adapted to robotic solutions for outdoor risky interventions.

1 Overview

Most papers concerned with real-time embedded application design present experimental tests showing that theoretical results, obtained from formal analysis, match real-time behavior of embedded system. But they do not consider critical aspects of integrating, or interfacing, with other non-real-time compatible processes such as complex high-level control system or user-end applications. To override the complexity of computing such application algorithms, the control software is built using a dedicated software environment: iCORE. iCORE relies on the SynDEx[1] data flow graph formalism (introduced by INRIA) which objective is to provide rapid prototyping and error free implementation procedures for distributed and heterogeneous application.

From this flexible and reliable development approach, we present how our advanced mobile systems could be easily used and customized for implementing outdoor applications, and guaranteeing integrity during risky interventions. Each point of the method is mainly discussed through one of the Robosoft transportation products: the robuCAB. It is a car-like mobile platform designed specifically for urban applications as well as automated transport. Last, in order to illustrate the possible robot operating mode, we will focus on software modules covering various needs: autonomous navigation, fleet management, remote control, specific HMI...

2 iCORE development environment

Robotics solutions, we describe here, make use of custom control architectures (composed of one Intel x86 Linux/RTAI machines and from one to 8 Motorola MPC555 based control boards[2]) with CAN buses as communication media. This section covers both the development and embedded targets environment.

---

1 Synchronized Distributed Executive

2 cb555 boards manufactured by Robosoft as part of his own control system products

2.1 iCORE: an approach based on SynDEx methodology

The application development method discussed here makes use of both SynDEx tools and Robosoft kernels. Developed by INRIA, SynDEx V6 is a graphical interactive software (see Fig. 1) with on-line documentation (refer to [3]), implementing the AAA[3] methodology. Here is the list of services offered by the couple SynDEx and Robosoft proprietary kernels:

- specification of an application algorithm as a conditioned data-flow graph (or interface with the compiler of one of the Synchronous languages ESTEREL, LUSTRE, SIGNAL through the common format DC)
- specification of a multi-component as a graph
- heuristic for distributing and scheduling the algorithm on the multi-component with response time optimization
- visualization of predicted real-time performances for the multi-component sizing
- generation of dead-lock free executives for real-time execution on the multi-component with optional real-time performance measurement. These executives are built from a processor-dependent executive kernel. SynDEx comes presently with executives kernels for various digital signal processors, microprocessors and micro-controllers.

The distributing and scheduling heuristics as well as the predicted real-time diagram, help the user to parallelize his algorithm and to size the hardware while satisfying real-time constraints. Moreover, as the executives are automatically generated with SynDEx, the user is relieved from low level system programming and from distributed debugging. This allows optimized rapid prototyping and dramatically reduces the development cycle of distributed real-time applications.

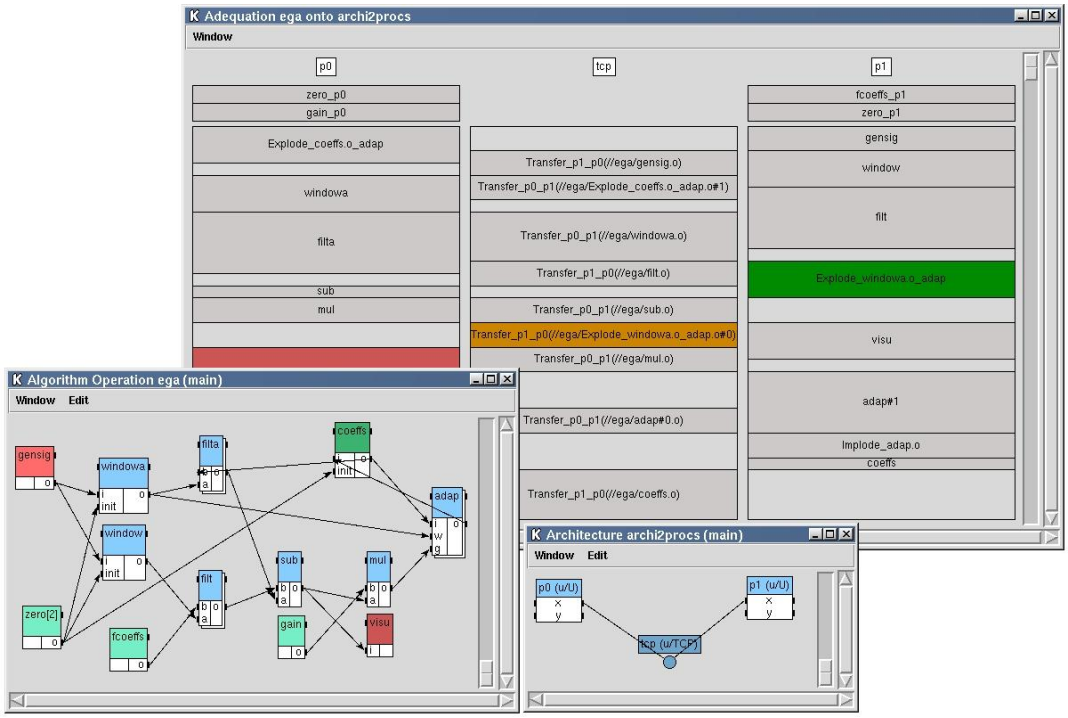

Fig. 1: Application design example using SynDEx CAD

3 Algorithm Architecture Adequation

The SynDEx development system described above is able to generated executive binaries for various types of target including the ones that compose the Robosoft control platform. Robosoft control architecture typically embeds one or more MPC555 based boards and an Intel x86 Real-time Linux computer.

2.2 Robosoft cb555 control board

The Robosoft cb555 [4] board (see Fig. 2) is a stand-alone four axis controller designed for critical industrial process handling. Including a 32-bit PowerPC architecture, it provides high computation performance (refer to Table 1 for a detailed description of boards IO capabilities).

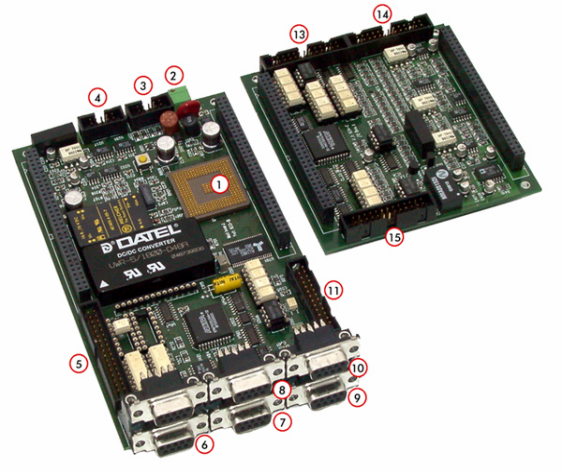

Fig. 2: The cb555 Robosoft control board

Figure label	Description
1	MPC555 microcontroller
2	Power supply: 18-60V DC (from batteries)
3	BDM Interface (Basic Debug Interface)
4	7 analog inputs
5	16 logical inputs and 20 logical outputs
6	Synchronous serial line (SPI)
7	Asynchronous serial lines (port 0)
8	Asynchronous serial lines (port 1)
9	CAN bus (port 0)
10	CAN bus (port 1)
From 11 to 14	connectors dedicated to axis control (including 1 analog output per axis)

Table 1: Robosoft control board connectors description

2.3 Robosoft emPC and wsPC computers

The context of real-time embedded applications programming is quite different from the classical one, user usually meets. This notion of "real-time" is not present in normal Linuses. Such real-time dedicated mechanisms can be added by installing an RTOS[4] on top of Linux standard kernel [1] [2]. Robosoft based both emPC and wsPC product ranges (resp. for embedded and workstation computers) on RTAI version which is widely used in embedded industry for prototyping and which is supported by very active companies.

RTAI basic principle is rather simple. RTAI provides deterministic and preemptive performance in addition to allowing the use of all standard Linux drivers, applications and functions. To this aim, RTAI decouples the mechanisms of the real-time kernel from the mechanisms of the general purpose Linux kernel so that each can be optimized independently and so that the real-time kernel can be kept small and simple. In our case, the primary function of RTAI kernel is to provide real-time tasks with direct access to the raw hardware, so that they can execute with minimal latency and maximal processing resource, when required.

3 The robuCAB implementation

The robuCAB platform (see figure 3) is a mobile robot offering great transportation capabilities, able to be driven over urban environments (such as cities, campuses...). Consequently, this platform perfectly fits a wide range of missions: autonomous transportation, tele-operation, fleet supervision... The platform control is handled by a heterogeneous architecture composed with both cb555 boards and an embedded PC. Figure 3 shows how hardware units are organized: two cb555 controllers dedicated to low-level critical loops, while the embedded PC focuses more on advanced algorithms and interfaces with asynchronous devices such as wireless network, GPS, laser scanner... Real-time communications sequences between cb555

[4] Real-Time Operating System

boards and PC rely on a CAN bus, while higher level communications (toward supervisors, network databases, web server, as well as any other non-real-time devices) are realized trough classical Ethernet links or serial lines.

Fig. 3: The robuCAB platform

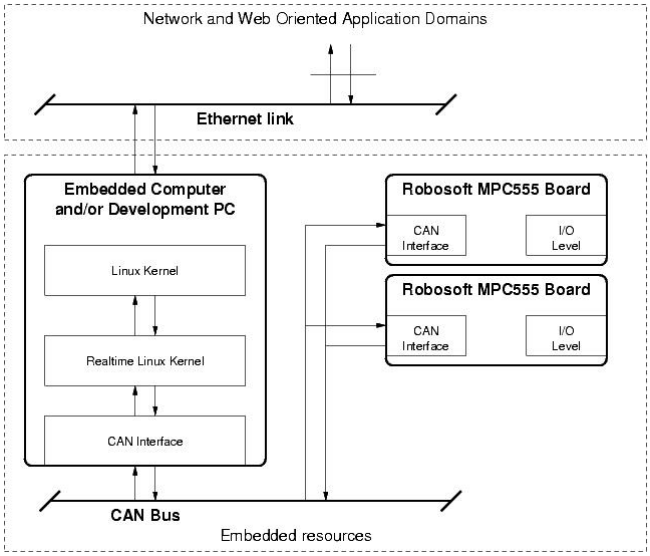

Fig. 4: robuCAB control architecture

Thanks to a very modular hardware and software structure, robuCAB can handled a wide set of application. As an illustration, let us focus on figure 5. It represents the structure of a supervised transportation application. Several software levels are depicted, from the most critical 1kHz task to the fully aperiodic web oriented processes:

- 1kHz control loops driving the four independent motors and the two independent steering jack servos
- a 100Hz loop dedicated to speed and steering profile computing

- an asynchronous level (running under Linux) handling both DGPS and LMS[5]
- aperiodic supervision processes with both web page and database update (running over one or more computers connected to a network)

Relying on iCORE environment (merging the best of SynDEx programming methodology and Robosoft modular proprietary kernels), these software levels implementation lead to highly predictable and safe application executions, as well as safe (non-blocking) interactions between software levels.

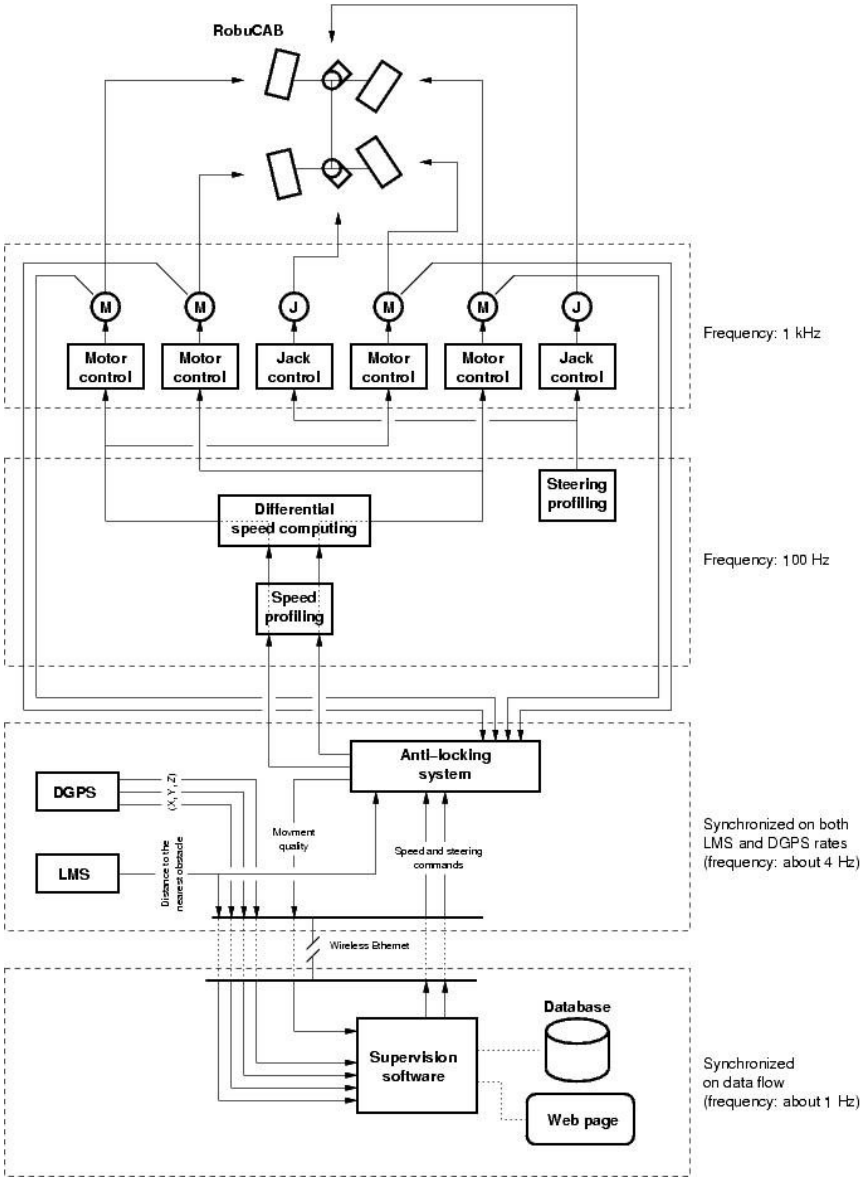

Fig. 5: robuCAB application structure

4 iCORE approach: a decisive step to both hardware and software modularity

Over the robuCAB example, we show that fairly complex and exhaustive software applications may be implemented, covering all the needs required for transportation missions. The iCORE approach, we

5 LMS SICK laser scanner

propose, appears to be very well adapted to robotics software development and, moreover, bring a very interesting modular concept. With iCORE approach, modularity is twofold.

First, relying on SynDEx methodology, iCORE provides users with hardware modularity. This means that once an application is written for a given architecture (composed with a set of cb555, PC and CAN buses), it is able to run on an extension of this architecture without any modification. Hence, extending the computing capabilities of a robotics platform is totally effortless: application is automatically redistributed in order to handle the new architecture resources.

Secondly, iCORE approach also provides user with software modularity. As shown on figure 1, in our context, an application is designed as a block diagram. Each block contains either a basic feature (from iCORE kernels) or another set of blocks implementing a more complex feature. Thus, iCORE approach make software part be easily reusable: simply by copying and pasting blocks subsets.

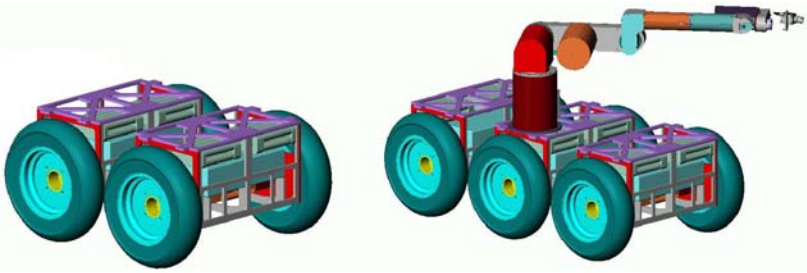

Fig. 6: Example of possible platform modularity

Figure 6 gives a striking illustration of possibilities offered by iCORE modularity. On the left side, a four-wheeled platform is shown, composed with two pods. Each pod is driven by the same piece of control software (running on its own cb555). Hence, programming control for a six-wheeled version (with three pods) is nothing but duplicating a diagram subset and adding a new cb555 into architecture graph. Code distribution and execution will require no other step. The same way, assuming a 6-DOF robot arm control has been previously realized using iCORE approach, adding such an arm to the 6-wheeled platform is nothing but merging the two application diagrams together and modifying the architecture description in order to fit the right set of cb555 boards and PC.

Conclusion

Relying on this flexible and reliable development methodology, the iCORE approach, we present here, bring an efficient help to users and researchers, interested in overriding application implementations complexity. Thanks to his own experience, Robosoft has adapted robotic solutions for the field of automatic transportation of people and goods, leading to a dedicated product range: rugged hardware components (cb555 and various types of embedded PC), as well as a set of realtime software components (control loops, I/O primitives, laser and wire guidance, obstacle detection, ...). As shown over the given examples, iCORE approach allow user to easily implement and customize transportation applications. Finally, making use of programming methodology introduced by SynDEx, iCORE guarantee applications executions integrity during risky interventions.

References

[1] E. Bianchi, L. Dozio, P. Mantegazza.
DIAPM RTAI Dipartimento di Ingegneria Aerospaziale - Politecnico di Milano
Real Time Application Interface.

[2] RTLinux
The Realtime Linux.

[3] Thierry Grandpierre, Christophe Lavarenne, Yves Sorel.
Modèle d'exécutif distribué temps réel pour SynDEx. 1998..

[4] MOTOROLA SEMICONDUCTOR TECHNICAL DATA.
MPC555 Product Preview PowerPC TM Microcontroller, MOTOROLA INC., 1998.