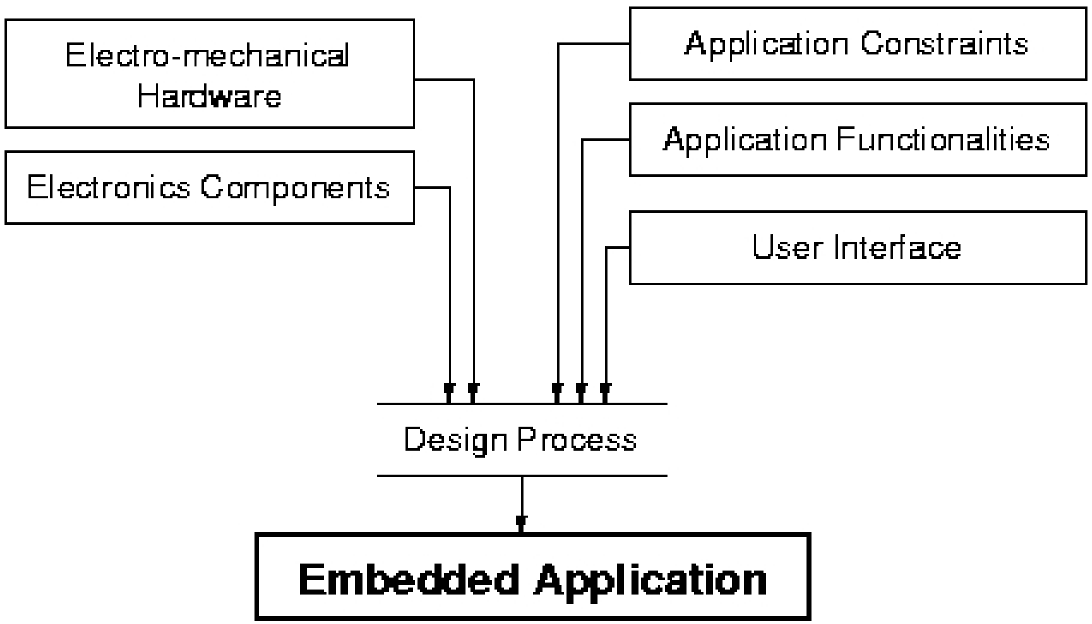
CAR'06

Modular distributed architecture for embedded robotics systems

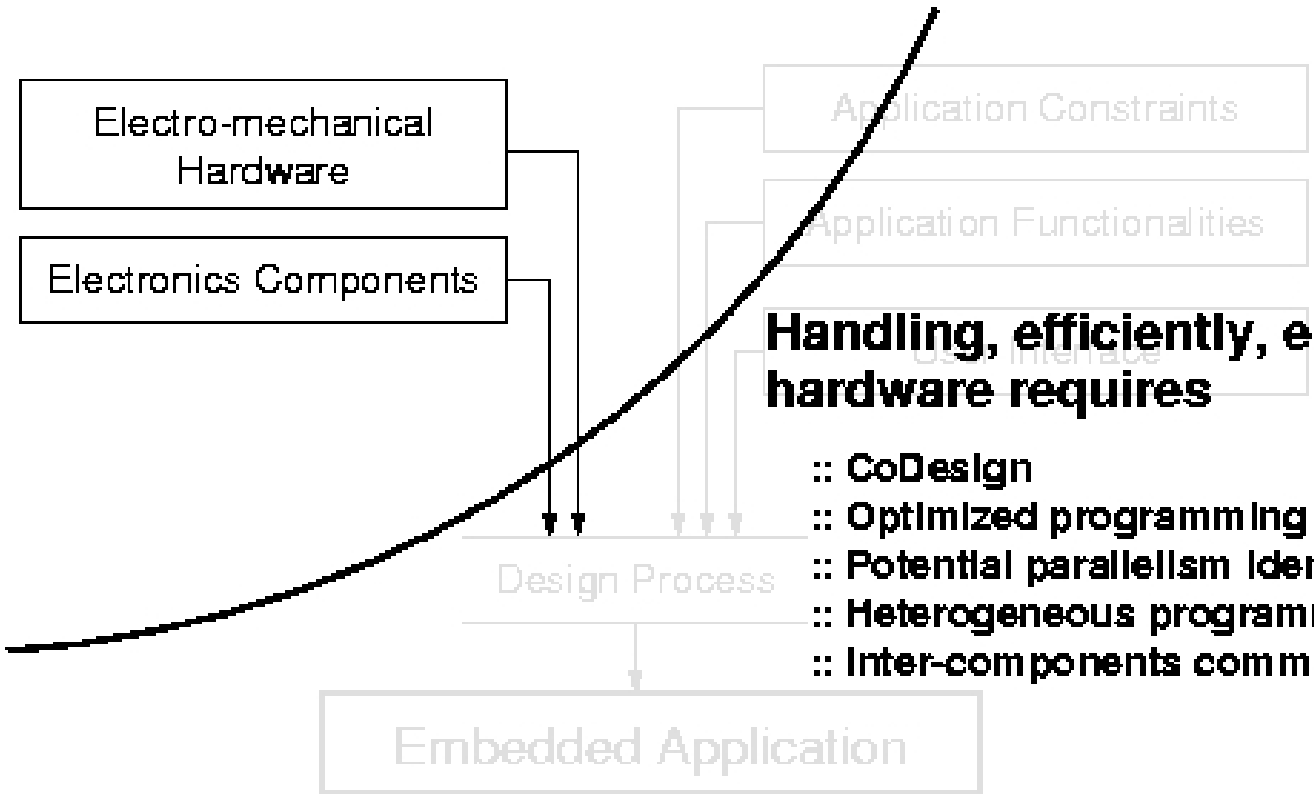
Pierre Pomiers <pierre@robosoft.fr>

::1-0 Implementing Embeded Applications

Implementing embedded applications deals with both system **hardware** and system **functional** layer.



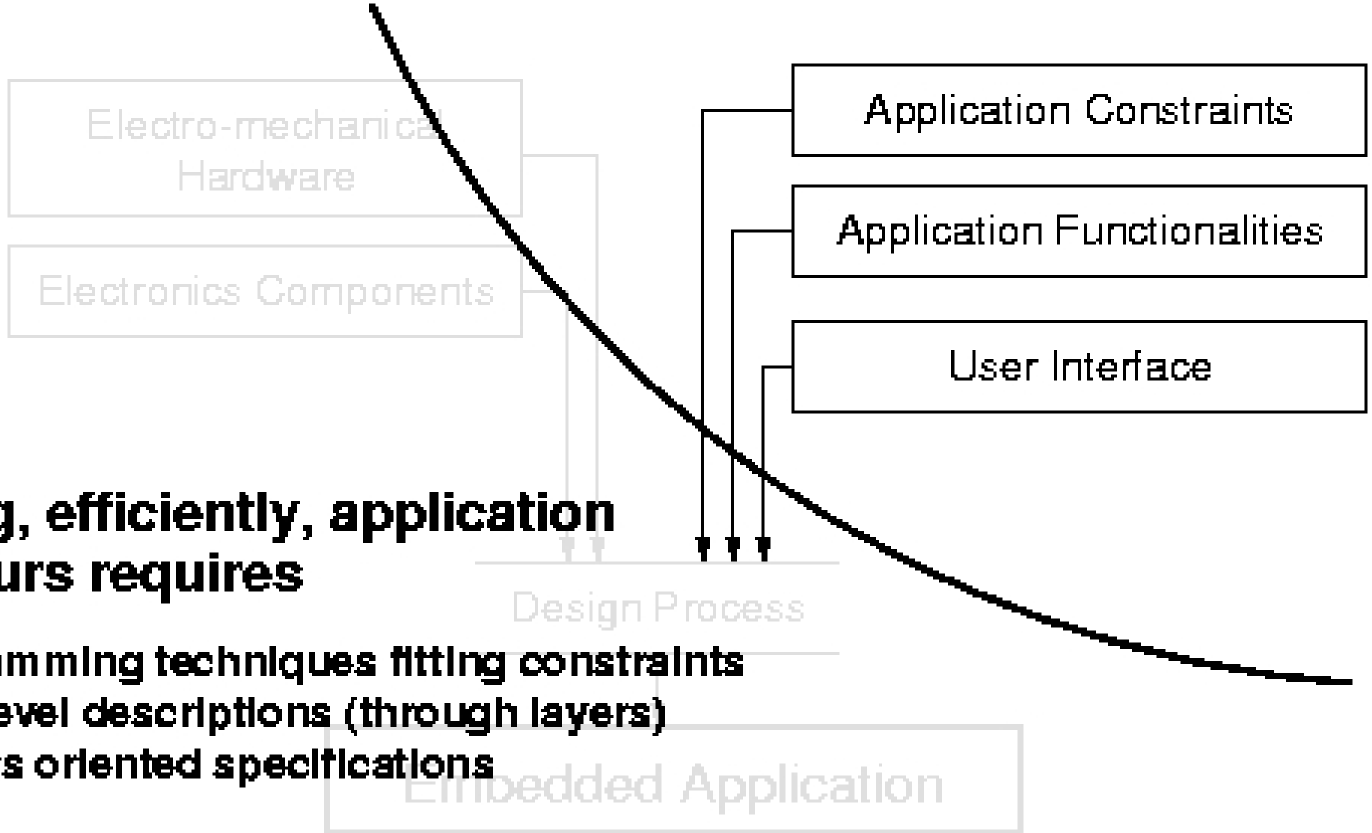
::1-1 Hardware Related Implementation



Handling, efficiently, embedded hardware requires

- :: CoDesign
- :: Optimized programming
- :: Potential parallelism identification
- :: Heterogeneous programming
- :: Inter-components communications

::1-2 Functional Layer Implementation



Handling, efficiently, application behaviours requires

- :: Programming techniques fitting constraints
- :: High-level descriptions (through layers)
- :: Objects oriented specifications

::1-3-0 Implementation Issues

Synchronous programming techniques are known to satisfy many **Hardware** related implementation points...

:: Automatic code generation (assembly, C...)
:: Automatic code distribution over the hardware architecture
:: Easier heterogeneity handling
:: Easier parallel and pseudo-parallel programming
:: Code verification (through synchronous formalism)

... but do not offer enough flexibility for higher level computing (that does not fit sequential finite state automata representation).

::1-3-1 Implementation Issues

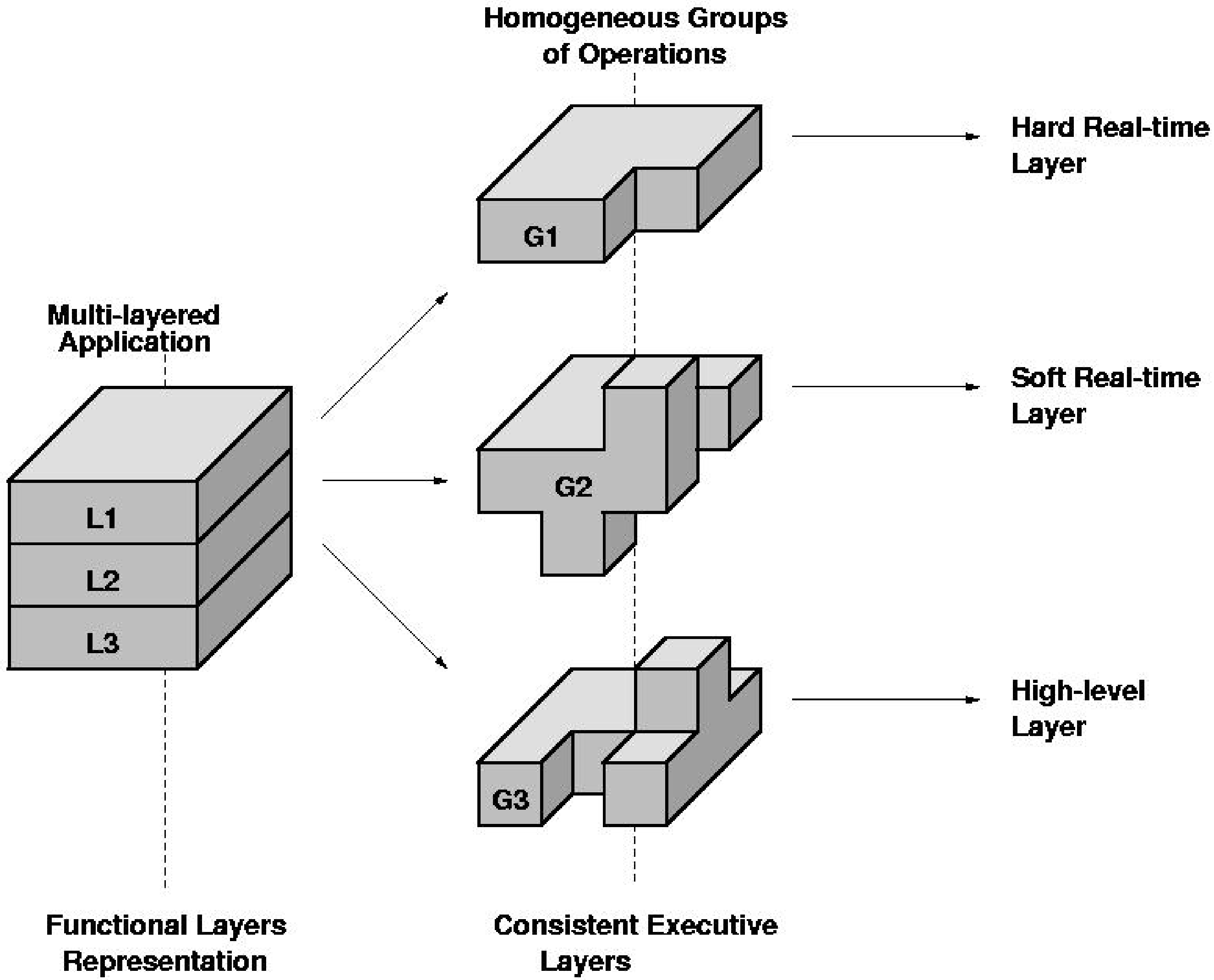
Concerning **Functional** layer programming, many approaches are satisfactory so long as they provide user with:

:: High-level representations (objects, tasks, processes...)
:: Multi-rhythm capabilities
:: Priorities handling and arbitration
:: Access to various OS services (networking, video...)

Nevertheless, such approaches are not optimized enough and can hardly satisfy requirements for embedded and distributed software.

::2-0 Introduction to The iCORE Methodology

Our idea is to mix up **both** high-level specification and, as far as possible, synchronous implementation. As well as proposing **well adapted hardware** for satisfying each layers constraints.



SynDEx key features::

:: **Rapid prototyping** of complex applications based on automatic code generation in three steps:

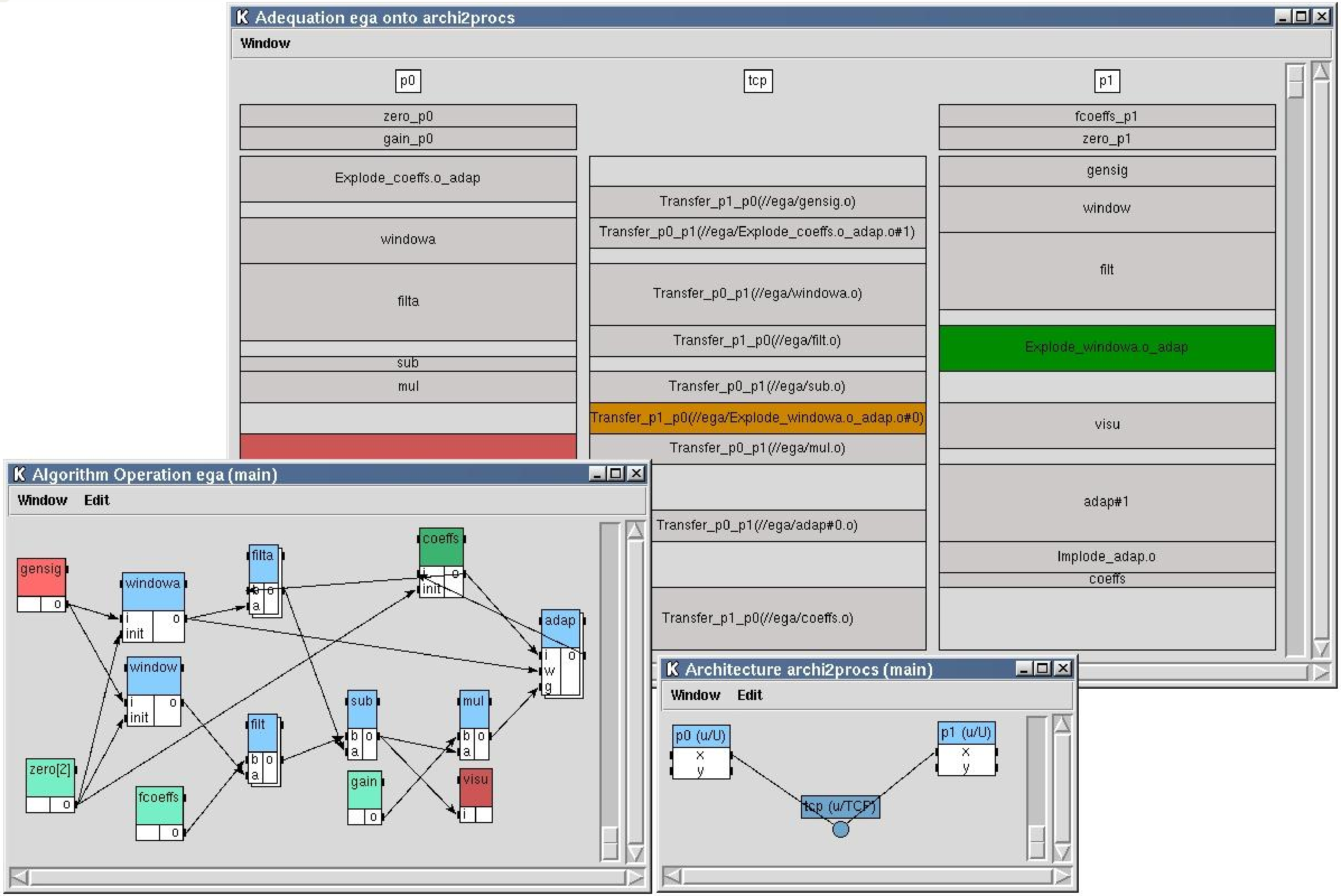
execution on a mono-processor workstation for simulation
execution on a multi-workstation in order to study parallelism benefits
real-time execution on the actual multi-component architecture

:: **Safety and Optimization** of multi-component real-time embedded code

:: **Co-design** when parts of the application must be implemented by software running on processors, while other parts must be implemented by specific integrated circuits.

:: **System level CAD tool** offering a software environment to help the user from the specification level (functionality, hardware resources, real-time and embedding constraints, of the application) to the embedded code level

SynDEx sample graph::

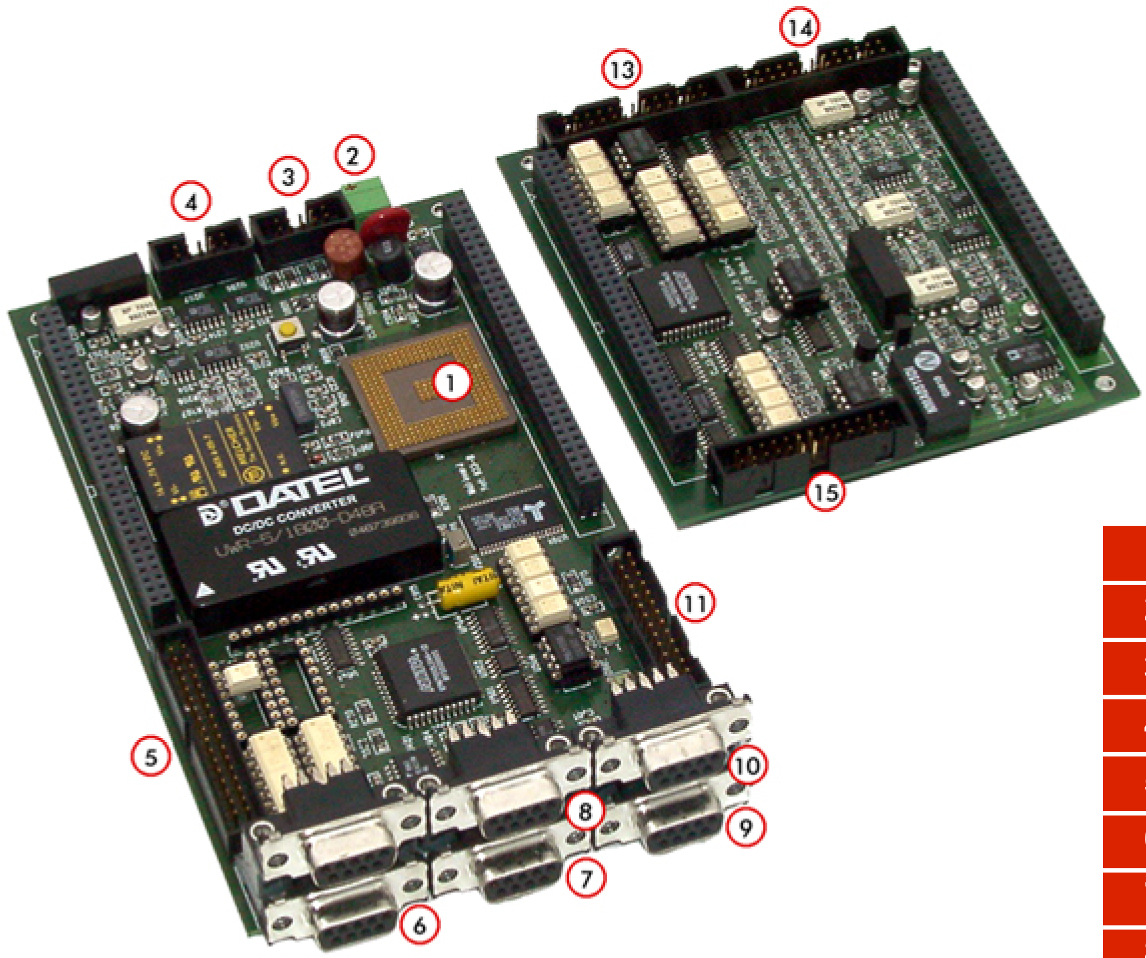


::2-2 iCORE Embedded Hardware

Thanks to **iCORE** kernels and cross-compilers sets, the SynDEx development system described above is able to generated executive binaries for various types of target and communication media:

- MPC555 based control boards
- Intel x86 Real-time Linux computer
- CAN bus controllers
- Serial controllers
- Ethernet controllers

::2-3 cb555 Control Board Features



	Description
1	MPC555 connector
2	Power supply: 18-60VDC (from the batteries)
3	BDM Interface (Basic Debug Interface)
4	Analog input (Potentiometer, Joystick)
5	Logical Input/Output
6	Synchronous serial line (Absolute Encoder)
7	Asynchronous serial lines (Port 0)
8	Asynchronous serial lines (Port 1)
9	CAN network (Port 0)
10	CAN network (Port 1)
11	Axis 0
13	Axis 1
14	Axis 2
15	Axis 3

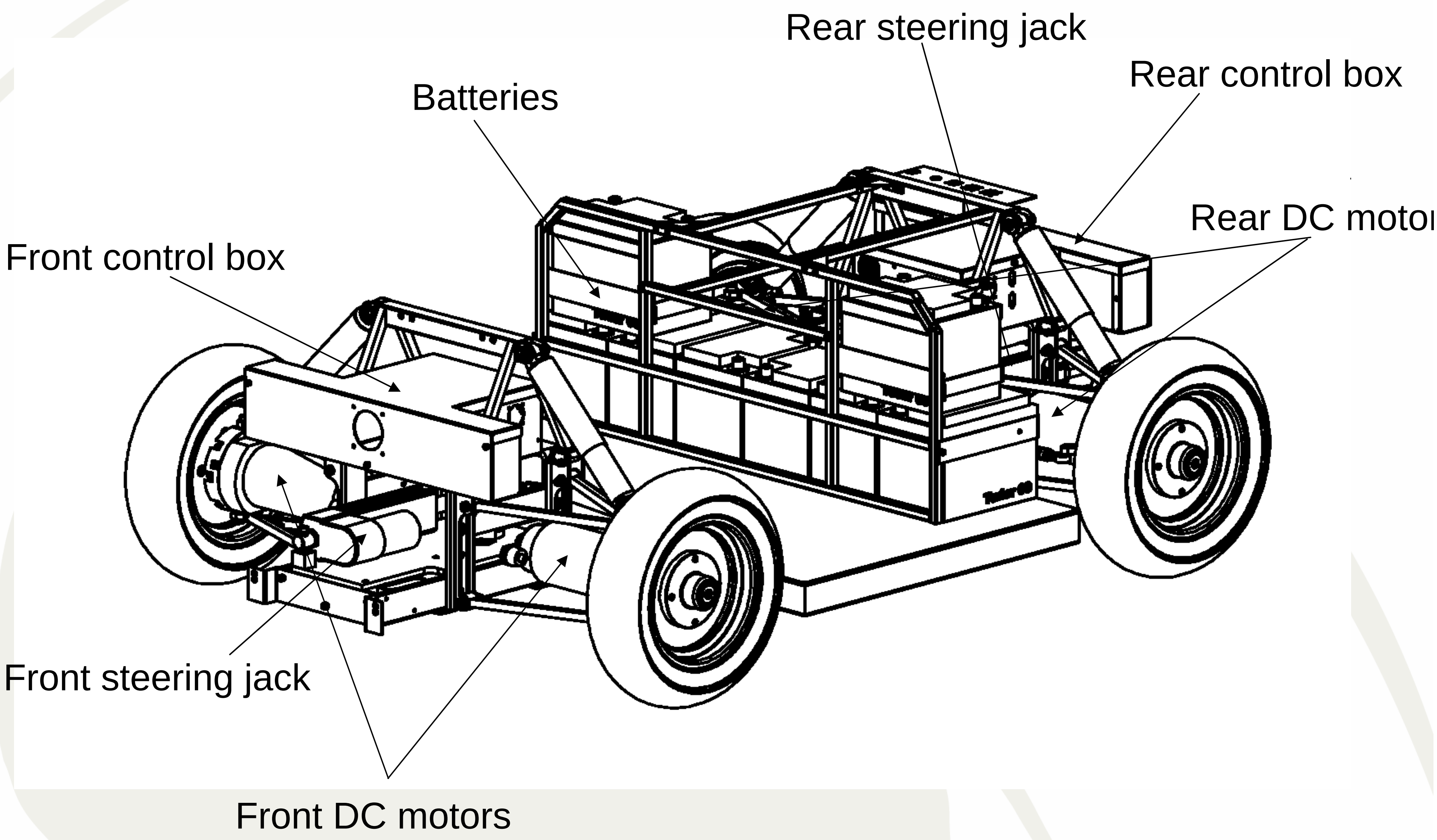
::2-4 emPC: Embedded PC Range

A wide range of embedded PC is supported (through Linux/RTAI ports): from Pentium-based **SBC** to **STACK104** industrial PC and more...

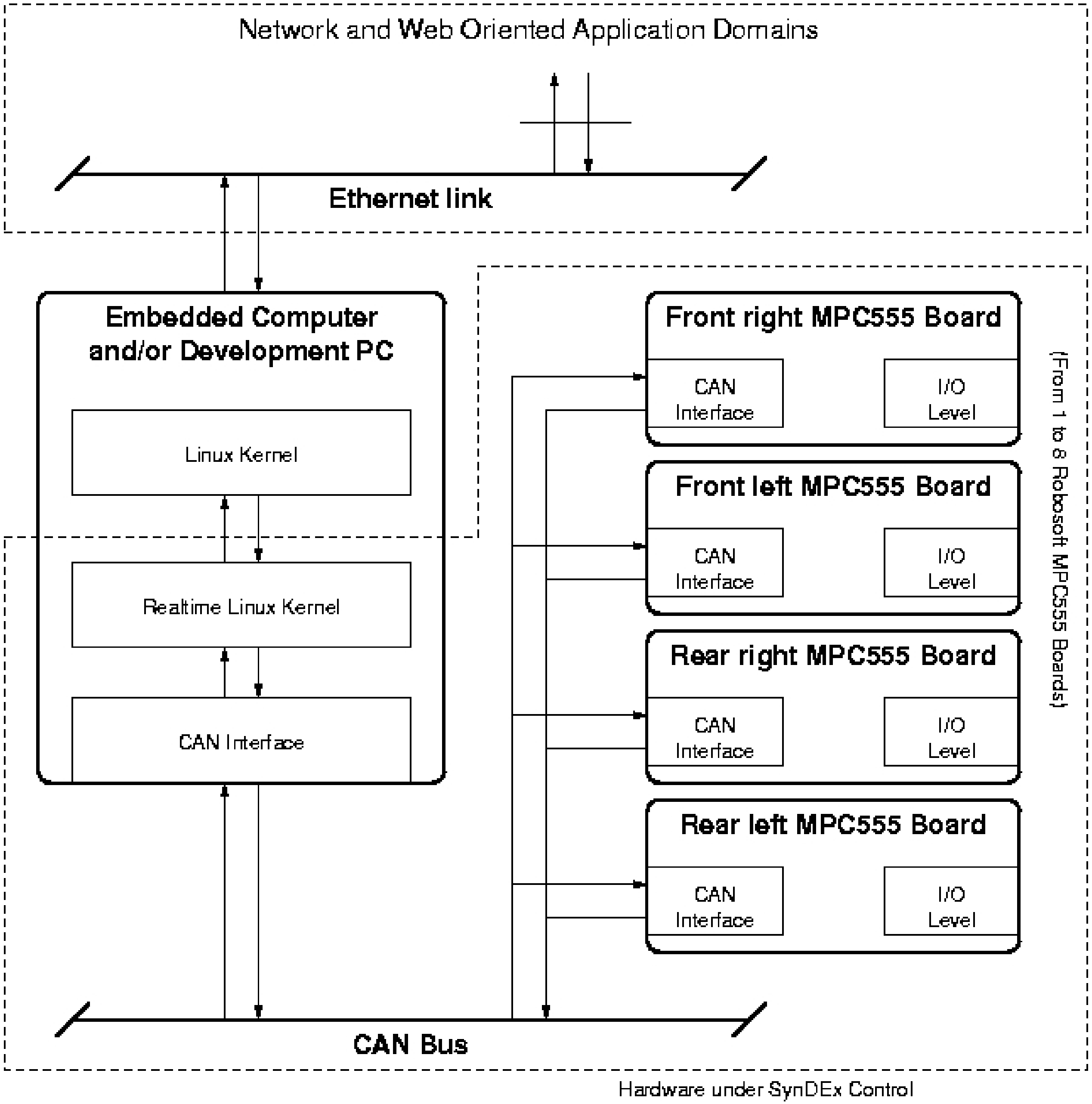
::3-0 The robuCAB Application

Let us illustrate the **iCORE**-based approach through the case of an all-terrain embedded application: the robuCAB

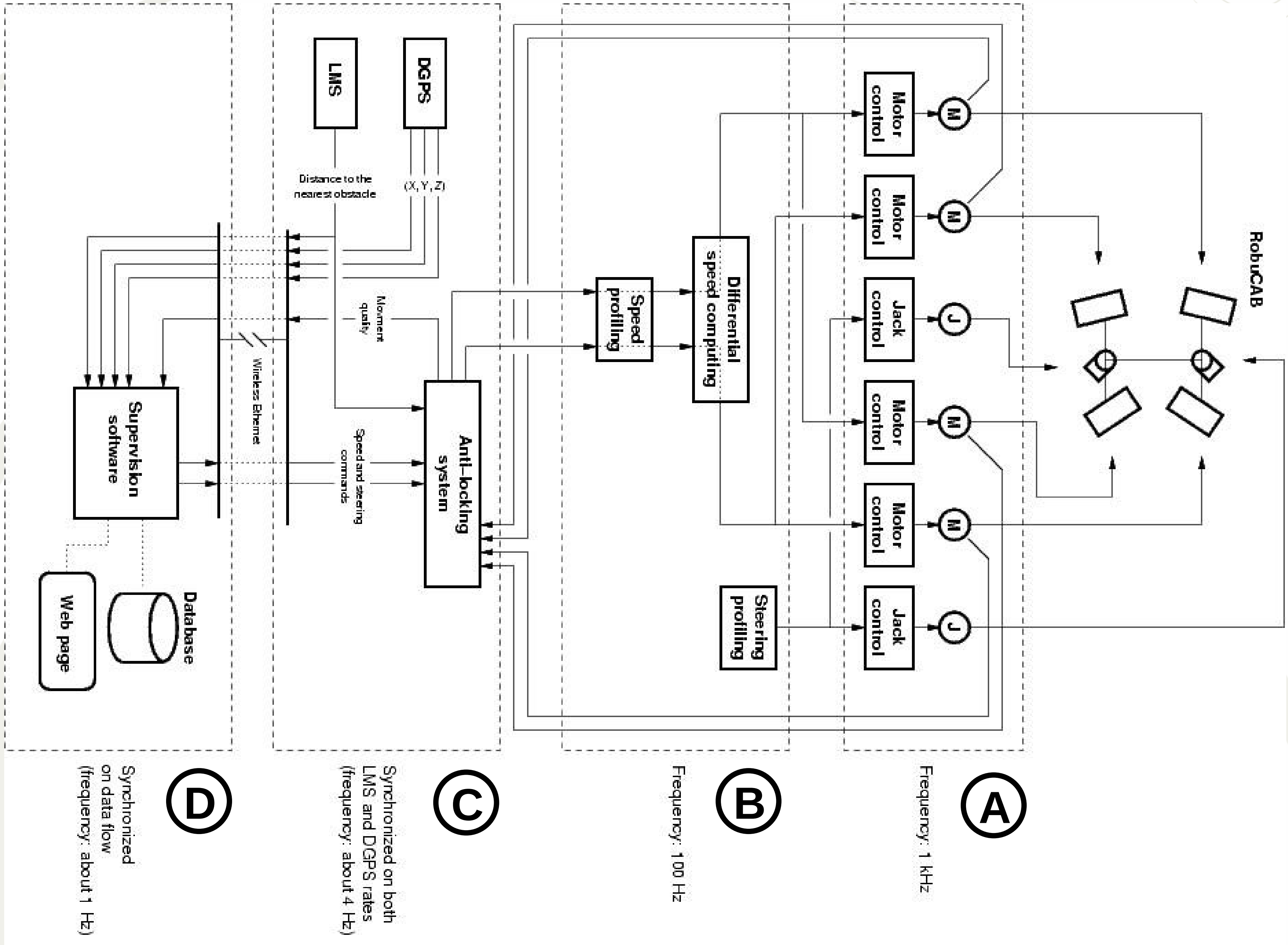
::3-1 The robuCAR Mechanical Structure



::3-2 robuCAR Hardware Architecture

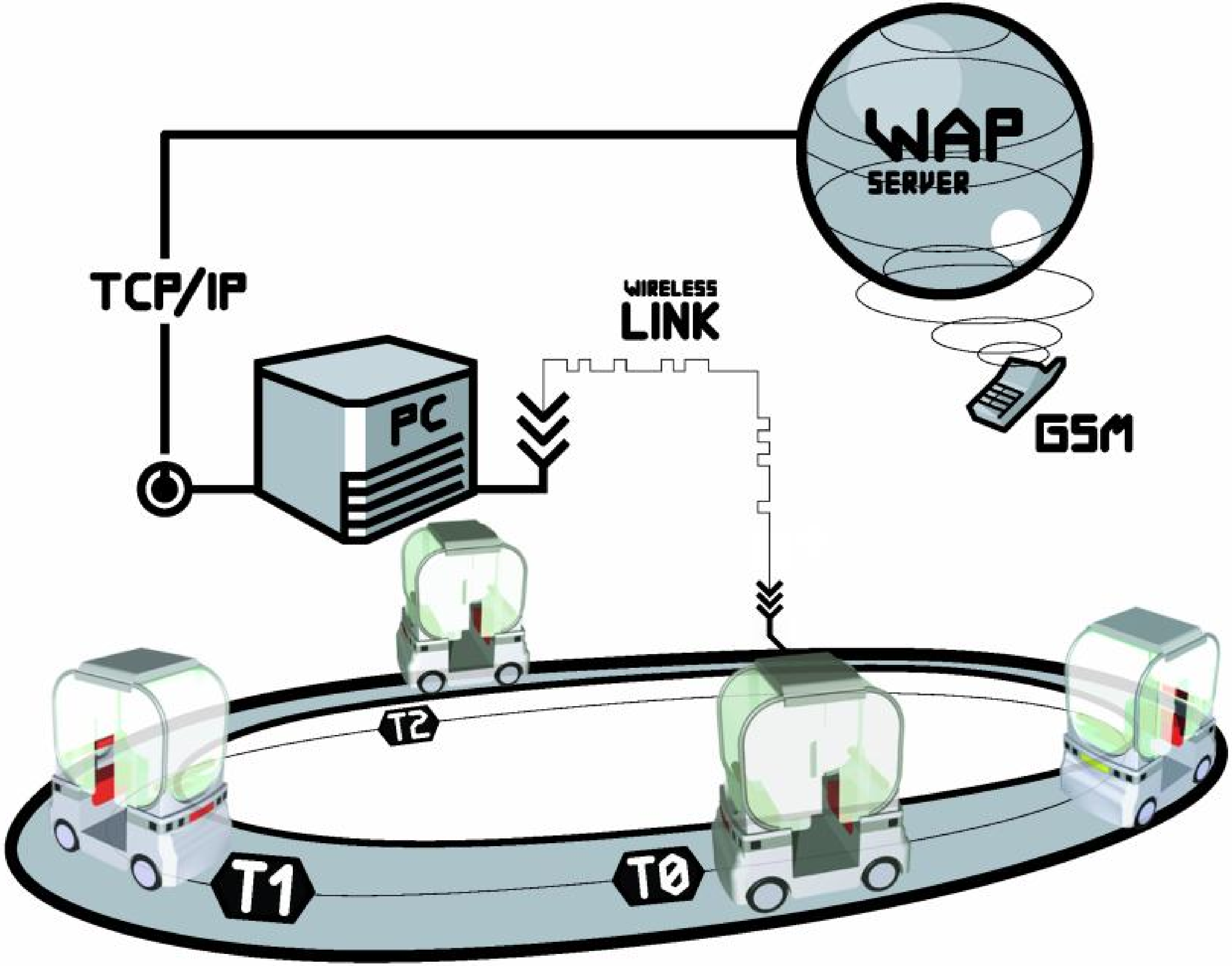


::3-3 The robuCAR Control Structure



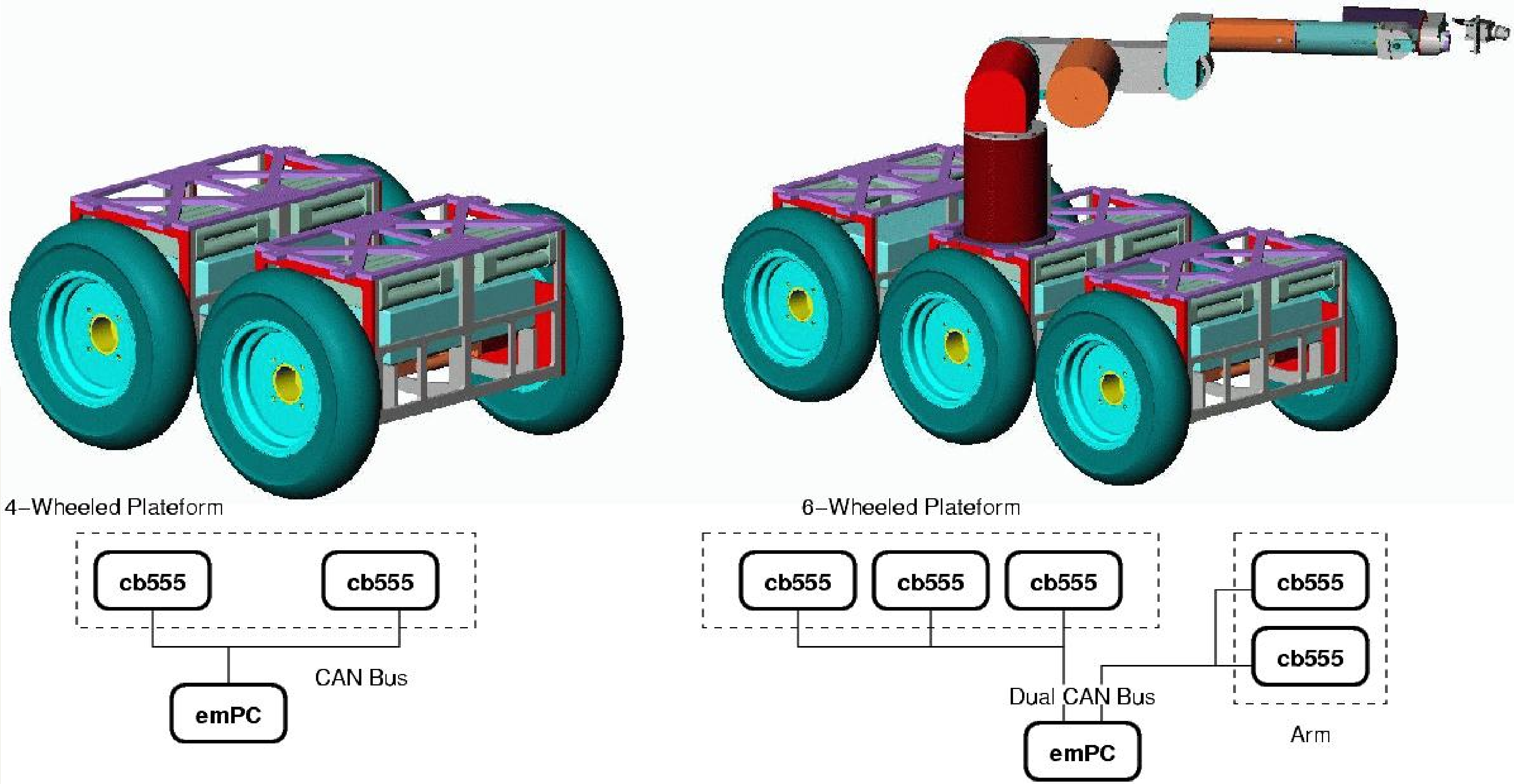
::3-4 Application Example

iCORE-based approach allow a wide range of software interactions.



::3-5 Application Example

iCORE-based approach allow flexible hardware design as well as modular software implementations.



iCORE key features:

- Rapid and safe prototyping of complex embedded and distributed applications
- Handle a wide range of software libraries (Linux/RTAI)
- Handle a wide range of hardware (cb555, emPC, ...)
- Merging robotics software modules is effortless
- Provides both hardware and software flexibility and modularity

::5 Contact Us

ROBOSOFT

Technopole d'Izarbel
F-64210 BIDART
Tel (+33/0)5 59 41 53 60
Fax (+33/0)5 59 41 53 79

http://www.robosoft.fr