ANNOUNCEMENT

Initiative to Apply Java™ to Automotive Control Systems

February 2000

Automotive Control Systems Today

Today vehicles include an increasing number of electronics systems. It has been estimated by Dataquest that the average semiconductor content of a vehicle will reach $240 by 2001, with consumption of DSPs, microcontrollers and microprocessors reaching $4.9 billion. Typical configurations consist of interconnected Electronic Control Units (ECUs) which may be connected through several buses. One possible bus is the "entertainment/infotainment" bus which connects subsystems such as radio, telephone, and navigation. Another one is the control bus which connects ECUs such as the ABS, the engine control, or the transmission control subsystem. A vehicle such as the Volvo S80 includes "18 ECUs connected via six networks: a low-speed body electronics CAN bus (125kbit/s), a high-speed powertrain CAN bus (250kbit/s), and four other networks."

ECUs are part of an embedded system market which is very fragmented since it ranges from systems based on 4 bit microcontrollers (e.g. a light switch) to 128 bit processors (e.g. game consoles). Today 4 to 16 bit microcontrollers make up the bulk of the market in terms of volume (around $10 billion per year or higher), while 32 bit and higher systems count for less than 10% of the market. It is however expected that 32 bit systems will expand at a much higher rate than 16 or 8 bit microcontrollers.

Most ECUs in the automotive control systems market are currently 8 to 16 bit systems. An increasing number of 32 bit systems will be included in the next generation of vehicles, but for cost and reliability reasons, such systems are likely to be designed using single chip or system on a chip approaches. This implies significant constraints in acceptable memory footprints.

ECUs often have to deal with duty cycles of several ms. Engine control subsystems for instance have to manage the engine cycle which has a duration that depends on the round per minute (RPM). At 6000 RPM, the engine cycle is 20 ms, which means that the ECU has to handle timing constraints on the order of several ms. In addition, ECUs exchange control data through the bus. The exchange period can be as low as several ms. The CAN bus is a typical type of control bus currently in use. Note that in many cases, engineering decisions made by OEMs may add further real-time constraints, such as the implementation of a serial link or a multiplexing link by software to lower cost.

ECUs are typically designed and developed by OEMs according to requirements set up by car manufacturers. Such requirements include a detailed description of expected interfacing capabilities, and in particular a detailed description of data transmitted to the ECUs or to be transmitted by the ECUs. In many cases, OEMs are not informed of the entire vehicle message system, as the car manufacturer may wish to protect trade secrets.

Until quite recently OEMs had in most cases total freedom in the design of ECUs, including the software, the processor and the hardware. However car manufacturers now anticipate a need to provide very precise requirements, in particular at the software level. There is a trend toward the specification of more global functions made possible by the networking and multiplexing capabilities of today's vehicles, for example an automatic gear shift function could sit partly in the transmission control ECU and partly in the engine control ECU. Thus a car manufacturer may wish to subcontract an OEM for the development of only the software component (e.g. the global gear shift function or the part sitting in the engine control) or for the production of an incomplete ECU (e.g. the engine control part without the automatic gear shift part). This approach not only allows the implementation of global functions, but it can also further protect the car manufacturers trade secrets.

To address this trend, the automotive industry is looking for technologies that will facilitate the software development integration process. It has identified the need to provide guidance on the transition towards advanced electronic architectures in the following ways:

- Pushing from the start for the definition of open systems, with the definition of corresponding interfaces (APIs). This has allowed the definition of the OSEK/VDX standard

(`www.osek-vdx.org`), probably the most successful undertaking concerning RTOS standardization in the software industry today. Currently there are more than 10 different OSEK/VDX providers and many more proprietary implementations from car manufacturers and OEMs.

- Pushing for the use of advanced software engineering methods, such as OMT and UML, and approaches promoting software reuse, such as object-oriented programming.

Java could be the cornerstone technology on which the infotainment bus could be built. The AMIC initiative (`www.ami-c.com`) on multimedia interoperability, created in October 1998 by five leading manufacturers (GM, Ford, DaimlerChrysler, Toyota, Renault) which now includes all car manufacturers, is investigating an open technology in the multimedia area based on Java.

On the other hand, Java has not yet been considered for automotive control systems because of technology issues, like memory footprints and real-time.

The AJACS Initiative

AJACS (Applying Java to Automotive Control Systems) is a two-year project partially funded by the European Commission. Its objective is to specify, develop and demonstrate an open technology allowing the use of Java in deeply embedded automotive ECUs such as engine control systems.

An open technology is being defined that will:

- rely on existing standards of the automotive industry, in particular OSEK/VDX,
- fully retain the benefits expected from object oriented language programming in terms of software structuring, reusability, dependability,
- fully retain the WORA (Write Once Run Anywhere) and robustness attributes associated with Java,
- support the same kind of real-time constraints which non Java based ECUs currently handle,
- target the type of electronic configurations that are in use in the industry: 16 and 32 bit microcontrollers and memory footprints ranging from 64 Kbytes to several hundred Kbytes (ROM) for the whole system.

Mechanisms and APIs are being defined to:

- support typical standards of automotive control systems, e.g. specific APIs for diagnosis, communication, and network management, support the OSEK/VDX standard, and coexist with OSEK/VDX based applications written in C,
- conform to future Java deeply embedded standards and Java real-time standards,
- address typical timing constraint issues for automotive control systems,
- address programming language issues such as
  - memory management issues (e.g. garbage collection, persistent objects),
  - synchronization issues (e.g. priority inversion),
  - support for asynchronous external events (e.g. support of interrupt handlers in Java),
  - support for asynchronous thread notification,
- address distributed communication and multiplexing needs (e.g. use of CAN, subsystem management),
- allow for the right level of CPU performance (e.g. native code generation), and allow the development of device drivers in Java.

AJACS work will is being carried out in close collaboration with currently available specification efforts (Java Community Process, J consortium, OSEK/VDX).

The partners of the AJACS project are:

- Trialog (OSEK/VDX technology provider)
- PSA (car manufacturer)
- CRF (technology center for Fiat, car manufacturer)
- Mecel (technology center for Delphi, equipment manufacturer)
- University of Karlsruhe.

AJACS project duration: 2 years.
Starting date: February

™ Java is a trademark of Sun Microsystems

TRIALOG

Software Engineering Focused on Embedded Systems Technology

AJACS Project Manager: **Antonio Kung, Trialog**
25 rue du Général Foy, 75008 Paris, France
Tel: +33 1 44 70 61 00
Fax: +33 1 42 94 80 64
e-mail: antonio.kung@trialog.com
www.trialog.com