Safety, Efficiency and Autonomy - Mastering Conflicting Trends in Embedded Systems Design

Rolf Ernst

Technische Universitt Carolo-Wilhelmina zu Braunschweig
Institute of Computer and Network Engineering
Hans-Sommer-Str. 66,
38106 Braunschweig, Germany
r.ernst@tu-bs.de

Extended Abstract

Embedded systems have developed from single microcontrollers to networked sys-tems and are moving further on to large open systems. As an example, automotive electronics started as a single microcontroller for engine control to develop into a local network of 50 and more electronic control units connected via several network standards and gateways which are found in current cars. These networks will be ex-tended by open wireless car-to-car or car-to-infrastructure communication enabling completely new functionality, such as advanced driver assistance systems that report approaching cars that could cause an accident. Other examples are found in health-care, where patients are monitored at home connected to a hospital data base and monitoring system rather than staying in the hospital for that purpose, or in smart buildings where different control functions are integrated to minimize energy con-sumption and adapt consumption to the available energy, or in energy supply net-works that are optimized to include renewable energy production. In all these cases we observe a transition from local closed networks with a single systems integrator controlling all design aspects (such as an automotive manufacturer) to larger open networks with many independent functions and different integrators following differ-ent design objectives. The Internet plays an important role supporting that trend. Unlike closed networks with a defined topology, such systems change over the life-time of a system.

As a consequence, there is no single design process any more that controls all components and subsystems. There is no single "product" that is replicated in produc-tion, but every open networked system is somewhat different both in implemented services and in topology. Updates and upgrades change the system over its lifetime. Lab test and maintenance become increasingly difficult as neither execution platform nor system function are fully defined at design time. Many deeply embedded nodes are hard to reach or become so large in their numbers that a centrally controlled maintenance process becomes infeasible. To handle such challenges, autonomous, self learning and evolutionary system functions have been proposed which automatically adapt to changing environments

M. Hinchey et al. (Eds.): DIPES/BICC 2010, IFIP AICT 329, pp. 5–6, 2010.
© IFIP International Federation for Information Processing 2010

and requirements. Unfortunately, this reduces system predictability which is a main requirement to guarantee system properties such as real-time and safety.

A second consequence is the convergence of system functions with different de-pendability and safety requirements. Patient monitoring at home is certainly a safety critical task that runs in a home environment that was intended for home office, entertainment and home appliances with lower safety requirements. So, if we want to use the IT environment at home for monitoring, it must be able to handle higher safety requirements. A similar requirement holds for car-to-car communication if safety critical driver assistance functions shall be implemented this way. A future traffic assistance system is likely to include pedestrians and bicyclists using their mobile devices to communicate with cars and warn of hazardous situations. This will be particularly helpful for senior persons. Now, the mobile device and its communication channels will become safety critical which is a completely new requirement.

This host of conflicting requirements is likely to become a showstopper for many advanced embedded applications if system and service providers are not able to give guarantees and assume liability. One approach is isolation of resources. Most promi-nently, time triggered protocols and architectures have been proposed that assign unique time slots to each application in order to minimize side effects. This is a con-sistent but conservative approach which has a major impact on the autonomous development and evolution of a system. Unfortunately, current hardware components have a deep state space (caches, dynamic predictions) that affects execution timing beyond even longer time slots. That makes complete isolation in time rather difficult. Multicore based systems with shared resources are a good exam-ple for the upcoming challenges.

As an alternative or complement, formal methods have been proposed that analyze system properties, such as timing and safety. Today, they are typically used in support of embedded system simulation and prototyping, but in future autonomous systems they could run automatically since test cases and evaluation are not needed. First examples have been presented in research demonstrating feasible computation requirements.

Even if the upcoming systems integration challenges can be handled with autono-my, suitable computer architectures, and formal methods, they will not be for free. Lack of cost and power efficiency could still prevent their introduction, in particular where energy resources are scarce. So, systems integration and control of autonomous embedded systems should be seen as a global optimization problem using a separate global control function, much like the control layer of a classical communication network, but requiring guarantees that are far beyond the current state.

at https//www.scipedia.com to download the version without the watermark