The Role of Standards in Real-Time Computing

Ulrich Schmid

Technical University of Vienna, Department of Automation (183/1),
Treitlstraße 3, A–1040 Vienna,
Phone: 0043-222/58801-8189, E-mail: s@auto.tuwien.ac.at

Standards have been defined in almost every traditional engineering discipline for a long time. Computer industries became aware of the importance of widely accepted standards particularly with the development of computer networks. Today, they are sometimes viewed as a panacea: there are standards for networking (e.g. IEEE 802.x), operating systems (e.g. POSIX), hardware (e.g. VME bus), etc. This is also true for real-time systems. In Europe, for example, there are numerous firms offering the development of customary real-time applications ranging from small embedded systems to geographically distributed factory automation and industrial process control systems. They usually combine a considerable experts' knowledge within some field of application with the use of industrial standard components. Actually, these industries would not be able to survive economically if such hardware (e.g. VME-bus CPU- and I/O-modules) and software standards (e.g. kernels like VRTX) were not available.

The major deficit of the industrial standards mentioned above, however, lies in the fact that many of them are built upon an insufficient conceptual basis: Does it actually make sense to worry about a RTEID (Real-Time Executive Interface Definition) or ORKID (Open Real-Time Kernel Interface Definition) standard while taking it as granted that static priority scheduling in conjunction with traditional interprocess communication facilities are to be used? Similar arguments apply to any real-time UNIX implementation, including the "celebrated" POSIX.4 (real time extensions) standard/draft, of course. Admittedly, the situation is particularly complicated for real-time systems due to the necessity of an integrated approach: Scientific research has pointed out that usual modularization concepts are worthless for real-time systems, unless both hardware and software components are integrated within a global scheduling concept.

Anyway, the actual problem with ill founded standards does not lie in applications built upon it. Instead, it is the "inertia" of already established standards against being replaced by novel ones (primarily resulting from the obvious desire to preserve existing hardware, software, and know-how), which causes serious problems. A typical example is the well-known Ethernet standard IEEE 802.3, which is based on a definitely unstable MAC collision resolution algorithm. In the meantime, much better ones have been developed (e.g. INRIA's Deterministic Ethernet), but nobody seems to be really interested in using them.

Therefore, it does not make sense to define standards for real-time systems at the moment — only because they are needed. Scientific research has to provide a sufficient conceptual basis first, before standards should be allowed to evolve.

This article was processed using the LaTeX macro package with LMAMULT style