Reliability, Safety, and Security in Everyday Embedded Systems

Philip Koopman

Carnegie Mellon University
Pittsburgh, PA 15213, USA
koopman@cmu.edu

Extended Abstract

Embedded systems permeate our everyday lives. From automobiles to elevators, kitchen appliances to televisions, and water heaters to cell phones, we increasingly depend upon embedded systems to operate as expected. A few obviously critical embedded application domains, such as aviation, have traditionally benefited from extraordinary care during development to ensure that everything is done correctly. But increasingly, everyday embedded applications are becoming “mission critical,” with little fanfare and perhaps without the full attention to dependability properties that they truly deserve.

Consider the following potentially significant failure modes for embedded systems: A cell phone that doesn’t work when the owner needs to call for emergency medical attention. A domestic hot water heater that overheats water, causing scalding burns on a child. A thermostat that doesn’t turn on heat when needed, causing household water pipes to freeze and burst. A microwave oven that turns on with the door open. An automobile that unintendedly accelerates. Today, hardware interlocks mitigate many of these hazards. But, software is playing a bigger role as both a vulnerability and a mitigation mechanism for critical failures. Because most embedded systems have actuators that influence the environment, and because people count on them to operate as expected, special care must be taken to ensure that they are safe, reliable, and secure.

Safety in the context of embedded systems deals with minimizing the frequency of mishaps (especially loss of life, injuries, and damage to property). In many ways this is the most mature of the areas we are discussing, because there are several industry-specific standards that can be followed to create safe systems (e.g., IEC 61508). There are, however, some significant research challenges outstanding in this area, including:

- How can we be sure that following a given system development process actually results in the hoped-for level of safety?
- How can we make it easy for small, non-specialist teams of domain experts to follow complex, “heavy-weight” safety standards and actually get it right?
- How can we simplify the representation and specification of safety properties to make it easier to design safe systems?

Reliability in embedded systems has been studied for many years, and has to do with ensuring that once an embedded system starts a “mission,” it has a high probability of completing that mission without experiencing a failure. Traditional high-reliability systems have used hardware redundancy (for example, two engines on an airplane instead of one). But, cost-sensitive everyday embedded systems often do not have a price structure that permits redundancy. An even bigger problem is creating highly reliable software, especially with quick time-to-market and low development budget constraints. Some current research challenges in this area are:

- How can we make it easy for small, non-specialist teams of domain experts to create highly reliable software?
- How can we quantify software reliability to support testing for design requirements such as “software crashes no more than once per month”?
- Achieving absolute software perfection seems unrealistic. How can we create embedded systems that survive the activation of latent software defects?

Security is, of course, a hot topic. But currently, it seems to be getting less attention in embedded systems than in enterprise systems. While embedded systems have not yet experienced as many widely publicized security problems as enterprise systems have, the potential for widespread, significant impact to society is certainly there. What happens if malicious attackers gain control of many embedded systems with the ability to release energy (or hazardous substances) into the environment? What if some critical infrastructure, such as energy distribution, traffic flow control, building environmental services, or telecommunications, suddenly stops working? While there are no easy answers to security in any environment, embedded systems present unique challenges that require research beyond the scope of enterprise security research, including:

- How can we make it easy for small, non-specialist teams of domain experts to get security right, even on a small product?
- What unique security challenges arise when interconnecting embedded systems (for example, coordinating actuators across many systems)?
- What novel vulnerabilities arise in Internet-connected embedded systems?
- What security concerns arise due to threats unique to embedded systems (for example, when the system owner is the attacker).

Embedded systems have historically been simple, often non-critical, and usually very reliable, safe, and secure. Newer systems are becoming more complex, and starting to cross the fuzzy line from non-critical to criticality. Unfortunately, the techniques and culture of developers for newly critical applications often do not take into account this major shift. While improving developer literacy in the areas of reliability, safety, and security will help, significant research challenges remain.

A common, underlying challenge has to do with the central role of domain experts in embedded system design. It is common for embedded system development teams to be relatively small, and staffed more with domain experts than computing experts. This is often appropriate, because expert domain knowledge is crucial to success. However, small teams and companies that are concerned mostly with an application domain rather than computer technology often don’t have access to expertise in dependability. So, even if researchers can solve the many outstanding research problems, there is still the issue of finding ways to deploy that knowledge to everyday working engineers whose training is often not primarily in computing. We must not only solve the research questions, but also find a way to deploy that knowledge.

This work was supported in part by the General Motors Collaborative Research Center at Carnegie Mellon University.