Transportation CPS Safety Challenges

Philip Koopman Michael Wagner
Carnegie Mellon University, ECE & Robotics, Pittsburgh, PA, USA
koopman@ece.cmu.edu; mwagner@rec.ri.cmu.edu

***Abstract*— Creating safe Transportation Cyber-Physical Systems (CPSs) presents new challenges as autonomous operation is attempted in unconstrained operational environments. The extremely high safety level required of such systems (perhaps one critical failure per billion operating hours) means that validation approaches will need to consider not only normal operation, but also operation with system faults and in exceptional environments. Additional challenges will need to be overcome in the areas of rigorously defining safety requirements, trusting the safety of multi-vendor distributed system components, tolerating environmental uncertainty, providing a realistic role for human oversight, and ensuring sufficiently rigorous validation of autonomy technology.**

Keywords—Cyber-Physical System (CPS) safety; ultra-dependable systems; software safety; autonomous vehicles

I. Introduction

A key concern in any Transportation Cyber-Physical System (TCPS) is safety. A variety of safety standards, building codes, and accepted practices help ensure safety for current transportation systems. [2] However, the increase in control authority and autonomy of TCPSs over time will mean that safety practices will have to evolve to keep pace. Accomplishing this will require addressing many research challenges, including coming to grips with the fact that autonomy algorithms are often difficult to validate to high levels of integrity, but need to be (almost) perfectly safe.

II. Historical Approaches To TCPS Safety

Most TCPSs are designed to be capable of safely carrying people. To that end, transportation systems typically have an allowable catastrophic failure rate in the neighborhood of 10^{-9} catastrophic failures per operational hour[1]. A "catastrophic" failure typically contemplates a mishap involving the death of many people (e.g., airplane crash with hull loss; major train derailment; multi-vehicle car collision). Less serious mishaps are permitted only a bit more frequently, generally limited to something like 10^{-7} fatal mishaps per hour (e.g., single-vehicle car crash). We refer to any system which has this general level of safety integrity requirement as a *safety critical system*.

A. Ultra-Dependable Failure Rates

The permissible failure rates for a TCPS are so low that they are difficult to grasp in the context of normal human experience. In more everyday units, a 10^{-9}/hr failure rate is one permissible catastrophic failure for every 114,077 years of continuous operation. Everyone who has had a computing device crash when they were using it has experienced a failure rate much worse than that. Everyday computing does not even begin to approach the needs of safety-critical computing.

The general technical safety strategy for most current TCPS systems is a fail-fast/fail-stop approach. In such an approach, redundant hardware components cross-check their operation to ensure fast detection of a run-time fault, and force the system to a safe state by shutting down the misbehaving system.

In elevators and trains, a fail-stop approach works well. Rail systems typically use additional redundancy to reduce outages by providing backup computers, but those backups are there to improve up-time, not safety. For cars, a fail-stop stalling of the vehicle engine is safe much of the time, although there are some cases such as accelerating onto a highway with no shoulder in which a fast shut-down can be problematic. It is currently assumed that failure of any automotive autonomy system leaves the car drivable by the human driver.

Aircraft are at the other end of the spectrum, requiring the aircraft to fail operational long enough to land safely at a diversion airport. However, even aircraft are largely built from individual components that fail-fast/fail-stop, with significant redundancy. (There is a reason commercial aircraft have at least two jet engines – one of them might shut down in flight.)

Redundant computing hardware is required in any safety critical system. If for no other reason, redundancy is required to detect faults caused by single event upsets from cosmic rays – and other problems – which occur orders of magnitude more often than 10^{-7}/hr. [3] Moreover, many faults cause system malfunction rather than a clean "crash," necessitating the use of cross-checking redundancy to ensure that failures are detected quickly, ensuring that the fail-fast assumption is valid.

B. Safety Critical System Validation

It is well known that testing is inadequate to ensure that safety critical systems are adequately defect-free. [1] Instead, a combination of approaches such as following safety standards, requiring system safety certification, and the establishment of a robust "safety culture" are relied upon to achieve acceptable levels of safety.

New aircraft designs require government certification before they enter revenue service, and are required to follow government-adopted safety standards for computer-based system safety (e.g., DO-178c). Railway systems are typically required to follow international safety standards (e.g., EN-50126/128/129) by the governmental organization purchasing

[1] Permissible failure rates are just discussion examples, but are broadly representative of the numbers given by standards in [2].

the equipment, with independent acquisition consultants often providing safety oversight. Both aircraft and rail systems have a well-established history of successfully ensuring safety, with safety problems notable by their infrequency.

Elevators must conform to government-adopted building codes that generally ensure safety via electromechanical means rather than certification of control software. Elevator safety is typically ensured by a combination of consultants and building inspectors who test those electromechanical safeties. In other words, elevators thus far have largely avoided the need to deal with software safety head-on.

At the other end of the spectrum, automobiles are required to conform to safety standards that largely do not get into the details of software safety concerns (e.g., Federal Motor Vehicle Safety Standards and Regulations – FMVSS). A previous-generation software safety standard [4] was adopted by some, but not all, automobile manufacturers. Enforcement of software safety practices varies considerably, with some companies using external safety auditing agencies. A more recent automotive software safety standard, ISO 26262, may be adopted by at least some automotive manufacturers.

Current TCPS safety tends to assume that all components have been integrated into a vehicle system by the same manufacturer, or integrated into a multi-vehicle transportation system by a combination of equipment manufacturers and contracting consultants working to a well-defined set of system-level specifications. External interfaces that affect safety, to the degree they are present, are carefully controlled.

A significant TCPS safety issue is the potential for problems caused by an ill-behaved external environment. Elevators and automated light rail systems physically isolate their operating tracks. Manned heavy rail systems, cars, and aircraft rely upon a human operator to ensure safety in the event of a malfunction or unusual operating situation. To accomplish this, they must ensure that the human can in fact take control of the system when the automation is incapable of ensuring safe operation. Transfer of control can be a bit thorny for situations in which the automation is attempting to prevent the operator from making a mistake, when it is in fact the automation that is incorrectly dealing with an exceptional situation. (This problem can be expected to get worse as more safety-critical TCPS functions are automated.)

III. TCPS Challenges

As computers subsume an increasing fraction of the workload for operating a TCPS, new challenges emerge that can fundamentally change the nature of the requirements for CPS functionality, dependability, safety, and complexity.

A. *Defining What "Safe" Means*

It is important in discussing TCPS safety to distinguish between two categories of safety.

Many TCPS safety discussions address adding functionality to a system to reduce system-level operational hazards, recover from operator errors, or address other problems at that level. For example, a safety feature in a car might be automated collision avoidance. These features might well improve safety – if they work as designed.

A different TCPS safety issue is the safety of the software that is implementing features. For example, a collision avoidance feature of necessity has the ability to control brakes, throttle, or steering (or perhaps all three). Such a function might operate incorrectly, causing a vehicle to crash during unusual operating conditions due to a software defect. The safety of a feature's implementation needs to be taken into account when assessing the benefit of adding safety features. In other words, a safety feature may improve safety when it works properly, but might be dangerous if it works improperly. Ensuring that safety functions are effectively defect free (i.e., no important software bugs) cannot be taken for granted.

Beyond these two types of safety, there is a challenge in rigorously defining what it means for a TCPS to be "safe." Traditional systems often end up defining "safe" as "obeys all specifications perfectly." This may well be too onerous a requirement to meet practically in a complex, autonomous, system. Moreover, there is usually no need for the system to operate perfectly to be safe. Much system functionality optimizes performance (e.g., fuel economy, ride quality, transportation capacity), and need not work perfectly for safety. Moreover, TCPS safety involves temporal aspects of the system, such as time shut down a vehicle after a component failure, and predicting whether a collision will occur at some point in the future. Therefore, it seems likely that temporal logic expressions of safety will be useful. However, it will be important to develop approaches that are accessible to non-specialist domain experts if they are to be adopted and used.

B. *Safety System Isolation and Trust*

Any transportation system has some features that are safety critical (e.g., speed control and steering control), and some features that are not safety critical (e.g., passenger entertainment). However, it is common for both types of systems to share system infrastructure such as communication networks or even computational platforms. Within a vehicle there are challenges to ensure that critical functions remain sufficiently isolated from non-critical functions, although some techniques such as memory protection are provide at least partial isolation in these areas. In a unified CPS model approach, it will be similarly be important to ensure that there are no sneak paths that enable non-critical portions of the system to undermine any safety-critical isolation assumptions made when synthesizing safety-critical portions of the system.

For multi-vehicle coordination, isolation may be more problematic. On the one hand, each vehicle's computational hardware is physically isolated from other vehicles, which helps. However, exchange of network messages creates security vulnerabilities that traditional TCPS designers are unaccustomed to handling. Malicious attacks on transportation systems via external networks are not just likely, but inevitable.

There are more subtle problems with coordinating behaviors beyond overtly malicious attacks. For example, how do two vehicles from different manufacturers trust that the other vehicle will operate in a safe manner? Even if strong cryptographic authentication and integrity are provided, that just tells Vehicle A that Vehicle B is really the type of vehicle it says it is. But how does Vehicle A know that Vehicle B's manufacturer actually designed it in a safe way, or that it doesn't have unsafe counterfeit replacement parts installed? For example, Vehicle B might report "I'm going to stop at this intersection; it's OK for Vehicle A to proceed through it," but due to buggy software or a substandard internal chip, Vehicle B will fail to actually stop. It might be that vehicles not only need to ensure secure communications, but also have a way to authenticate each other's safety certification credentials – down to the component level – before trusting each other.

C. *Environmental Uncertainty*

Traditional transportation systems relegate dealing with environmental uncertainty (e.g., deer crossing a highway) to the human. Alternately they provide controlled operating environments (e.g., automated platform loading doors at a light rail station). Operating in a less constrained environment will make it more difficult to ensure system safety, since it is difficult for designers to predict all exceptional situations that real-world systems could experience.

The transition to from partially to fully autonomous operation is likely to be the most difficult for cars, trucks, and shared-roadway light rail systems. While initial operation on a limited access roadway provides some level of environmental control (deer notwithstanding), operation on urban streets poses an extremely challenging environment if a human is not counted on to provide continuous oversight.

D. *Role of Human Operator*

As a system transitions from human-controlled to fully autonomous, there is a difficult transition region in which the system only needs a human to intervene in very rare occasions. This means the human may lose proficiency in operation that is needed if the automation fails. Or, the operator may simply be distracted (or even asleep), and not be available to instantly take control of a vehicle in an emergency.

Even if the vehicle operators remain attentive, it is essential that the system remain operational enough after an autonomy failure to give the human enough time to re-engage control and recover to a safe situation. For example, if a car is following a leading vehicle at a distance safe for an autonomous system, but too close for normal human reaction time to brake in an emergency, tossing control of braking back to a person just as the leading vehicle's brake lights go on makes it difficult for the human to avoid a crash.

E. *Autonomous System Algorithms*

Autonomous systems based on perception and control techniques from the field of robotics [5] present a new set of validation challenges for high-integrity operation.

Historically, autonomy algorithms have accepted that control decisions are unlikely to be perfect due to the difficulties in dealing with a near-infinite combination of operating environments, object geometries, object trajectories, and sensor noise. As an example, consider the performance of pedestrian detectors for cluttered urban environments. It seems exceedingly difficult to reduce false-negative rates to meet a requirement of, for example, 10^{-7} serious failures per hour (e.g., missed child in a cross-walk) without incurring an unacceptably high number of false positive stops at empty crosswalks.

Many of the challenges with autonomy algorithms can be thought of as stemming from their use of inductive reasoning approaches. For example, machine-learning techniques extract characteristics about classes of objects based on training data. Implicit in this is an assumption that the training data realistically represents all real-world objects. Moreover, there is a potential issue if environmental variations or sensor noise lead to incorrect conclusions about object characteristics (for example, whether a perceived bump in the road is just a bump – or a protected alligator basking on the warm pavement).

IV. Addressing the Challenges

Based on this discussion, some of the biggest challenges seem likely to come in the following areas:

- Specifying safety for a fully autonomous CPS;
- Ensuring safety for multi-vendor vehicular systems;
- Providing safe, fully autonomous operation in relatively uncontrolled, uncertain environments such as urban roads;
- Providing a realistic role for human oversight in almost-completely autonomous vehicles; and
- Attaining extremely low catastrophic failure rates in systems that use autonomous algorithms.

Addressing these challenges will require at least some combination of further maturing safety standards, education, and tool-based support for safety design not just at the vehicle level, but also at the level of complete transportation systems.

Acknowledgments

The authors wish to thank the General Motors Collaborative Research Lab at Carnegie Mellon University for more than a decade of research support on the dependability and safety of automotive systems. This paper is also benefits from long-term interactions with collaborators who design elevators, rail equipment, and aviation equipment. These opinions are those of the authors, not our research sponsors.

References

[1] Butler & Finelli, "The infeasibility of experimental quantification of life-critical software reliability,", IEEE Trans. SW Engr. 19(1):3-12, Jan 1993.

[2] Herrmann, Software Safety and Reliability: techniques, approaches, and standards of key industrial sectors, IEEE Computer Society Press, 1999.

[3] Mariani, Soft errors on digital computers, Fault injection techniques and tools for embedded systems reliability evaluation, 2003, pp. 49-60.

[4] MISRA, Development Guidelines for Vehicle Based Software, November 1994.

[5] S. Thrun, W. Burgard, and D. Fox. Probabilistic Robotics. MIT Press, Cambridge, MA, 2005.