NITRD/NSF National Workshop
on
The New Clockwork for Time-Critical Cyber-Physical Systems

October 25-26, 2012

Baltimore, Maryland

DRAFT Workshop Report

Draft as of Oct 11, 2012

Introduction

Cyber-physical systems have become central to our national infrastructure. Continuing improvements in cyber-physical systems will be crucial to improvements in all sectors of society, including health care, manufacturing, agriculture, transportation, and defense. For example, intelligent vehicles and highway infrastructure hold promise of maintaining safety, while handling larger traffic flows while reducing energy consumption and pollution. It is critical that these systems be safe, reliable, and trustworthy, yet affordable to develop and adaptable to changing needs. As these systems are expected to do more and more, they are becoming larger, more distributed, and more complex. As we entrust them with greater responsibilities, the penalty for failure becomes greater. Traditional ways of designing, constructing, certifying, and maintaining these systems are no longer adequate.

Most cyber-physical systems are critically dependent on correct timing. They need precise information about the times at which events occur in the physical world, and precise control over the times at which the system performs actions that interact with the physical world and other cyber systems.

Beyond time, they must keep track of relationships between space and time, for multiple physical entities. For example, intelligent highway vehicles must track and manage the movements of other vehicles and potential obstacles, as well as their own moving parts, as a function of time.

Achieving predictable operation of these systems with respect to timing, as well as other requirements, is made difficult by the need to operate in complex, dynamic, uncertain, and incompletely understood environments.

Of course, **there are limitations to what any system can do in an uncertain environment**. For example, an intelligent vehicle **cannot be expected to guarantee safety under all circumstances, but it needs to be as good as the best human driver**.

Societal and Economic Impact

CPS working in structure environments such as trains, and airplanes have greatly benefited the society in terms of safety and efficiency. However, this impact would be greatly expanded once these benefits are extended to unstructured environments such as regular highways, streets, and general uncontrolled airspace. In this case we can expect the benefits to include the following.

- Lives saved. Both by augmenting the human capacity to detect and react to dangerous situations and replacing them completely in some situations. The ability of computers to stay focused continuously and react at super-human speed would enable these systems to take corrective actions in critical situations. This would allow us to avoid accidents most of the time and/or reduce their consequences when such accidents are unavoidable.

- Increased efficiency of common infrastructure such as roads, railways, and airspace. This efficiency is derived from the combination of augmented capacity of human operators and fully autonomous functions in cars, trains, and planes. This capacity is translated into a more efficient operation that is traditionally limited by the human reaction speed requiring larger margin of errors (e.g. separation between cars). With adaptable CPS we would be able to reduce these margins, for instance, in terms of vehicle separation and speed increasing the density of the vehicles in the road leading to a higher throughput with smaller delays. This efficiency can be a very valuable capability in extreme situation such as the evacuation of geographic regions where natural disasters are expected (e.g. hurricanes or volcanic eruptions).
- Prevent economic disasters due to human errors in the operation of critical infrastructure. For instance, large-scale blackouts can be prevented by using CPS that can react faster than human operators that can both limit the effects of incidents in the electric grid and the duration of the incident (e.g. blackout). Similarly, the operation of other critical infrastructure such as water and sewer would be improved increasing its availability and safety, for instance, detecting dangerous conditions (e.g. contamination) and critical interruptions (e.g. pipe leakages/breakage).
- Robust and reliable systems that we can bet our lives on. In the end, the ultimate benefit is the capacity of having systems that, in the presence of uncertainty, we can trust with our lives. Some of these systems are already operational (roads, electric grid, water and sewer) or currently being built, but they do not have the properties presented here. As time goes on, the price we could pay for the absence of these properties will just keep increasing and can find us one day wondering why we never took care of preventing a nation-wide blackout, a water contamination incident, or the next massive accident in the one of our highways.

Modeling and Implementation of Time-Critical Systems

Introduction

Time-critical computer systems comprise an increasingly important part of the infrastructure that forms the underpinnings of modern society – witness the central role of time-critical systems in application domains as diverse as transportation (e.g., aviation, smart cars and smart highways), health-care (patient monitoring, tele-surgery, etc.), telecommunications, manufacturing, and the energy grid. As the scale and the complexity of these application systems continue to increase, it is no longer viable to depend upon ad hoc design techniques and the ingenuity of individual engineers and system designers to design, build, and maintain such time-critical systems, particularly given the catastrophic consequences that may arise from their failure. A rigorous methodology that covers the specification, validation, testing, implementation, and maintenance of large complex time-critical systems is needed, that is based on firm formal foundations and appropriate engineering practice, and is supported by easy to use tool-chains.

Societal and Economic Impact

As stated above, time-critical computer systems have proved of enormous benefit to society. As our dependence on them increases, however, the financial, social, and human costs of failures of these systems can be enormous – consider the following examples.

- Recent large-scale failures of the power grid have led to blackouts that leave millions without electricity (e.g., Southern California, September 2011).
- Failures of automated "algorithmic" trading programs have led to billions of dollars of financial loss (e.g., Knight Capital, August 2012, and the "flash crash" of May 2010 that caused the Dow Jones Industrial Average to plunge by about 1000 points in minutes).
- Repeated automotive recalls have been attributed to failures of time-critical software (e.g., 50,500 cars were recalled by General Motors over an airbag-related glitch, June 2011; a potential fire-hazard caused by problems with the power window control resulted in a million-car worldwide recall of Honda vehicles, September 2011). In 2010, 20.3 million vehicles were recalled in the United States.
- Between 1990 and 2000, over 200,000 pacemakers were recalled due to software issues. According to the FDA Infusion Pump Improvement Initiative, defective infusion pumps have been linked to over 19,000 serious injuries and deaths between 2005 and 2009. Software malfunctions are listed at the top of the reported problem list.

In addition to the direct consequences of failure, the need to avoid such failures currently carries a significant opportunity cost to society. Time-critical systems in safety-critical application domains are subject to mandatory certification; as these systems become larger and increasingly more complex, the cost of obtaining such certification is increasing exponentially with system size and complexity – in some safety-critical application domains, the cost of obtaining certification dominates the software development cost and probably discourages innovation. Once certification has been obtained for a product, there is a strong dis-incentive to improve the product since, lacking appropriate methodologies for incremental analysis, one would need to recertify the entire system once again.

The lack of rigorous methodologies for the design, implementation, and maintenance of time-critical systems also means that the process of developing and maintaining such systems is terribly inefficient in terms of time and human labor. Time-critical systems are often designed in an ad hoc manner that requires, for example, low-level programming that is tedious and cumbersome. Such practices divert expertise away from more interesting higher-level design and engineering challenges; further, the tedium of such work means that the best and brightest young minds are less likely to be attracted to the discipline of designing and engineering time-critical systems.

Conversely, having advanced capabilities in designing and implementing time-critical computer systems would provide US industry with a tremendous competitive advantage, particularly in safety-critical application domains including automotive and aviation.

What Can We Do Well?

Time-critical systems that we are currently able to design and implement in a satisfactory manner can be classified into three broad categories.

1. With considerable design effort, we are able to devise ad hoc designs for relatively simple safety-critical systems. These systems are typically simple enough that their run-time behavior (in terms of both timing and resource requirements) can be completely characterized, or in any event, bounded from above, at system design time. They are often implemented in a periodic time-triggered manner upon very simple, and hence highly predictable, hardware. If more complex hardware is used, advanced features of the hardware are not used: for instance, commercial off-the-shelf (COTS) multicore processors used in safety-critical systems typically have all but one computing core disabled in order to reduce non-determinism and enhance predictability.
2. We are also quite successful in building some complex time-aware systems, usually on non-distributed platforms, for which the consequences of failure during run-time are not particularly severe (and hence an inability to make a priori performance guarantees prior to run-time is not fatal). Examples of such successes are commonly found in the consumer electronics industry – consider the complexity of, e.g., smart-phones and the latest-generation video-gaming consoles.
3. Continuing advances in clock-synchronization technology have also made it possible to develop complex systems for real-time data collection. That is, it is now possible to time-stamp and collect large amounts of data across geographically distributed systems, with the time-stamps being consistent

across the entire system to a very fine granularity. However, latency issues in communicating across distributed platforms mean that it remains very difficult to build systems that are able to make control decisions based upon such collected data, in real-time.

What Do We Not Do Well?

Broadly speaking, while we have enjoyed considerable success in designing and operating timing models for complex systems such as smart grids, avionics, telecommunications and factory automation systems, the time synchronization and distribution architecture for each such system is highly specialized, has evolved within its domain silo and operates on specialized domain-centric platforms. It is hard to use the solutions from one domain more generally across newer domains such as medical devices or networked vehicles.

The essence of the challenge facing us may be stated as follows:

We lack a methodical process for correctly designing, implementing, and maintaining complex, time-critical systems in an efficient manner. ("Efficiency" here refers to both the resource-efficiency of the implementation, and to the efficiency, in terms of time and human effort, of the design and implementation process.)

Currently, each complex time-critical system is typically designed and implemented in an ad hoc manner. This requires considerable time, effort, and ingenuity on the part of the system builder, and the resulting implementation tends to make poor use of platform resources (including CPU computational capabilities, energy, etc.), and hence has a poor SWaP (Size, Weight, and Power) profile. Furthermore, it is extremely difficult to ensure correctness. In particular it is difficult to obtain a high level of assurance that the behavior of the implementation is compliant with design expectations – this is a particularly serious issue in safety-critical application domains that may further be subject to mandatory certification by statutory certification authorities.

In somewhat greater detail, several shortcomings of the current state of the art have been identified. These include the following.

- We lack appropriate formal models for representing, and reasoning about, time in cyber-physical systems. Our models of timing are currently CPU and network centric, but as systems get deeply integrated within physical substrates they need to better align with the timing requirements of physical processes. With deeper integration of closed-loop control and actuation of physical plant processes, integrated timing models are needed that can represent time-triggered, event-triggered and self-triggered operation of systems based on plant dynamics such as stability, energy-consumption, safety and performance. These integrated timing models should facilitate the preservation of timing properties across design, simulation/ testing, and implementation boundaries during the design and implementation process
- We lack adequate theories of composability for time-sensitive systems that allow us to reason about time properties of larger systems that are built from

smaller component systems. Amongst other problems here, we do not yet know how to identify, and deal with, unanticipated emergent temporal behaviors that are usually undesirable and may prove costly to deal with, that result when simpler time-critical systems are combined to form more complex ones.

- We find it difficult to deal with timed mixed criticality systems, in which functionalities that are characterized by different safety and/ or timing requirements (e.g., hard-real-time and soft-real-time) co-exist upon an integrated platform and therefore share platform resources.
- We lack tool-chains for designing and implementing time-critical systems that preserve time semantics from model-based (or even code-based) designs to physical implementation. Lacking such tool-chains, dealing with evolving system requirements, or changed (usually, improved) platform capabilities, becomes very difficult and requires significant redesign and reimplementation effort.

These shortcomings of the current state of the art motivate the discussion in the following section, on specific research challenges.

R&D Challenges

In general terms, we must achieve the capability to construct distributed processing platforms where time-dependent programs, which guarantee both safety and high performance, are synthesized from models that treat time is a first-class citizen. This requires interfacing modeling methodologies with real-time operating systems, programming languages, real-time databases, real-time middleware and network subsystems that can faithfully synthesize code for a large number of distributed computing elements and execute the synthesized time-critical programs without constraining the timing variability of the overall system. Several specific R&D challenges will help us achieve this broad goal; these challenges are individually discussed below.

Models and Representations of Time

Many time-varying signals of interest to cyber-physical systems – i.e., the signals that are monitored by the sensors of such systems – vary continuously with time; however, a computer system can only handle discrete representations of this signal, typically obtained by sampling. It is known that there are families of continuously time-varying physical signals that cannot be distinguished from one another, or from certain other discrete time-varying signals, by such discrete sampling. Moreover, it is known that uniform sampling is inefficient, and no form of sampling can unambiguously represent signals that are discontinuous. The theoretical and formal underpinnings of issues regarding the expressiveness of discrete-time sampling of continuously-varying signals are not well understood; further research is needed in order to better understand the implications of such discretization. In the absence of such an understanding, an ad hoc approach is introduced and adopted for each time-varying signal during the process of designing each individual system; such an ad hoc approach requires considerable effort and may result in non-robust systems that are difficult to modify in any manner, or to re-use all or part of one system design during the process of implementing a different system.

Composition and Interfaces

Complex systems are typically built by composing smaller, simpler components. Various theories have been developed concerning the functional properties resulting from such composition operations: given functional properties of the components, these theories allow one to determine the functional properties of the compound component formed as a result of the composition. Similar theories are needed for rigorously reasoning about the timing (and other non-functional) properties of composition operations. This will require the design of appropriate interfaces for components that hide un-needed details about the components (and thereby reduce the complexity of reasoning about their composition) while simultaneously exposing enough information via the interface to enable the derivation of the timing properties of the results of composition. This means that theories of composition must be developed that are "well-behaved" with regards to composition operations. For instance, permitted operations must be rigorously proved to not lead to unpredictable and undesirable emergent behaviors. In a similar vein, the composition of timing requirements is currently not well understood. Current techniques that are based on, e.g., determining the cross-product of the timing requirements of the individual components tend to not scale with system size and complexity, and are hence of little use in the design of large, complex, systems.

Another essential issue that has not been addressed in traditional theory on composition and interfaces is composition overheads for timed systems. For example, in non-timed systems the number of tasks does not affect the overall behavior; in fact, the semantics of concurrent systems are defined in terms of the equivalent sequential system of constituent components. For real-time systems, this technique of taking a product is inadequate, since the timing overheads due to task dispatching, cache contention, etc. are affected by the internal structures of the components.

There is a need for design techniques that are resource-aware and compositional; that is, they abstract away internal structure of subcomponents and operate only on the level of interfaces. A challenge here is to find appropriate abstractions for the components and efficient interface composition techniques that preserve the timing behaviors of components upon complex platforms, such as multi-core and distributed architectures, without adding too much resource overheads.

Resource-Efficiency Considerations

The problem of constructing large complex time-critical systems in a correct manner is challenging in and of itself. However, it is becoming increasingly important that these systems also be implemented in a highly resource-efficient manner. (This is due in part to energy and thermal considerations: even if plenty of computing capacity can readily be made available, providing the energy needed to enable all this computing capacity is fast becoming a bottleneck in complex modern systems. This problem is further exacerbated in mobile platforms that are not tethered to the power-grid. The related problem of heat-dissipation in order to prevent inadmissible increases in the temperature of the platform is also often a primary concern, as are considerations of the size and the weight of mobile devices.)

All these, and some other, considerations speak to a need for being able to implement time-critical systems in a manner that is both correct and resource-efficient. However, resource-efficient implementation of time-critical systems is an immensely challenging problem, and considerable research effort is needed to devise techniques and methodologies that realize this goal.

One particularly difficult challenge to obtaining resource-efficient implementations of time-critical systems arises from the increasing trend towards multicore CPUs; the best CPUs from the perspective of performance per unit cost, as well as energy efficiency, are currently multicore, and this trend towards multicore CPUs is only likely to continue and become more pronounced over time. However, meeting timing guarantees while executing upon multicore CPUs is extremely difficult: a great deal of research is needed in order to understand how to do this in a resource-efficient manner. Amongst other problems is the fact that many modern multicore CPUs contain highly sophisticated features (such as deep pipelining, multiple levels of cache memory, support for out-of-order speculative execution, etc.) that are optimized for delivering improved average, rather than worst-case, performance. Consequently provisioning resources upon such CPUs based on worst-case estimates results in extremely poor utilization of the platform resources, whereas provisioning according to average-case estimates may result in a failure to meet timing constraints. New models for specifying resource requirements of time-critical systems are perhaps needed, that are able to ensure timing constraints are met without needing to reserve resources on the basis of worst-case assumptions. A major research initiative on real-time operating system (RTOS) support for multiprocessor and multicore platforms also appears necessary, to enable them to provide the system support needed to implement time-critical systems in a resource-efficient manner upon inherently non-deterministic multicore CPUs.

In mixed-criticality systems, functionalities of different criticalities co-exist on a shared integrated platform. Examples include the computational infrastructure in a car that supports safety-critical functionalities such as braking, automated steering, cruise control, etc., concurrently with comfort (e.g., air conditioning) and entertainment functionalities that are not safety-critical. It is well known that pure "priority-based" resource allocation strategies for implementing such systems, that strictly prioritize more critical functionalities over less critical ones, make very poor use of platform resources since they do not exploit the timing attributes of either the safety-critical or non-critical functionalities. The challenge in implementing such mixed-criticality systems is to obtain implementations that are able to provide functional and timing guarantees at very high levels of assurance to the safety-critical functionalities, while simultaneously not reserving such an excessive amount of the resources that it becomes impossible to provide any guarantees, even at far lower levels of assurance, to the non-critical functionalities. Research is needed to determine strategies that would accomplish this goal.

Complex real-time systems are increasingly built by integrating components that are independently developed. The integration of such systems on a common platform brings significant challenges in simultaneously meeting the real-time performance requirements of multiple systems sharing computational and communication resources. One promising approach to reduce resource use is to promote better sharing using real-time virtualization that can provide predictable timing guarantee and isolation. In particular, real-time virtualization on multi-core platforms would enable large complex time-critical systems to be supported by real-time cloud computing systems.

Another approach to improve resource use efficiency is to develop new scheduling and analysis methods for highly dynamic and adaptive systems. The resource requirements of such systems can change dramatically during their execution, due to interactions with the external environment and/ or internal system failures. One challenge here is to predict changes at run-time and integrate the resulting effects on resource needs into the scheduling and resource management algorithms in an efficient and accurate manner. Another challenge is to ensure predictable timing guarantees during such changes.

Time-Cognizant Abstractions and Refinements

The typical process for designing and implementing complex systems is to use high-level abstractions to initially describe the functional, timing, and other requirements of the system, and then progress through a series of refinements that successively include more and more additional details, eventually yielding a physical implementation upon an actual platform (that hence includes consideration of all relevant detail). Such system-development processes are relatively mature and well understood for non-time-critical systems; however, processes of this form that preserve timing semantics are not known – it is crucial that refinements be devised that preserve timing semantics as well as functional semantics.

This problem of obtaining semantics-preserving refinements is rendered particularly challenging due to the fact that many widely used high-level abstractions tend to make significant simplifying assumptions about timing properties (e.g., synchronous reactive abstract models make the simplifying assumption that actions, such as the execution of code, take zero "logical" time). Refinements for abstractions of this kind have to assume the responsibility of reintroducing timing complexities. The design of new high-level abstractions that do not make un-needed simplifying assumptions about timing properties and that can include timing uncertainties to be introduced at lower levels as details are added through refinement should be explored. The use of such new abstractions, that include consideration of timing needs and properties from the very beginning of the design process, would greatly ease the problem of determining refinements that preserve timing as well as functional semantics all the way down from high-level design to final implementation.

Programming Models and Engineering Techniques

The research agenda discussed above is expected to provide a deeper understanding of the foundational principles of time-critical computing. It is imperative that these principles be incorporated into programming models and engineering techniques that will enable their use in designing and engineering actual systems. Engineering issues to be addressed include determining whether certain kinds of time-critical functionalities are better implemented in hardware (including FPGA's) and if so, what kinds of interfacing is needed between the hardware and software in order to maintain common semantic notions of time across the interface. It is also important to address issues of system evolution: what programming methodologies and what engineering practices yield functioning systems that are best able to deal with changing system requirements, and to exploit the availability of improved hardware and system capabilities.

Toolchains

Due to the complexity of many time-critical systems, it is important that tool support be provided for each stage of the process of designing and implementing such systems. Furthermore, the challenge of ensuring semantics-preserving refinements makes it imperative that tools for the different stages be integrated into tool chains that share common semantic notions of time. Building such tools and tool chains, and formally demonstrating their correctness, is extremely difficult, but the benefits of doing so cannot be overstated. Building an effective well-designed tool that does not require deep knowledge of the theory upon which it is based (and is hence easily used by application-domain experts who may not be particularly familiar with the theory of time-critical systems) is perhaps the most efficient means of transferring the results of academic research to industry.

It is likely that different tools based upon different formalisms will be more appropriate to different stages of system development. Tool integration efforts, therefore, should provide support in two directions. On the one hand, vertical integration will aim at the tools that are aimed at different design levels such as timing requirements, control system models, and platform architectures. On the other hand, horizontal integration will aim to bring together the tools that are applicable at the same level, but consider different aspects (such as size, weight, power, computation, network bandwidth) or employ different modeling and analysis methods (such as automata verification vs. real-time scheduling theory). On the vertical dimension, methods for bridging the semantic gap between different design levels, especially between the models and the implementation platforms, are required to guarantee correct timing behaviors of the system. On the horizontal dimension, there is a need for interfaces that enable transformations between different analysis models as well as the composition of analysis results obtained by the individual models.

Research Strategies and Roadmap

We believe that the most appropriate response to the research challenges identified above is to fund research efforts that directly address these challenges. In addition, we believe these meeting these challenges will be facilitated by initiatives that encourage two further objectives:

- **Community building.** The research and development community that needs to come together in order to solve the research challenges we have listed above is currently rather fragmented. Different groups of researchers, who attend different conferences, publish in different journals, and generally tend to consider themselves parts of different research communities, address different aspects of the problem; these groups should be encouraged to collaborate closely in order to meet these challenges. For instance, the synergy resulting from collaboration between real-time systems researchers addressing correctness issues and those addressing efficiency issues would likely be of immense benefit. In a similar vein, hardware and software aspects of time-critical systems are currently often considered separately; these, too, should be studied within a common framework by closely collaborating teams of experts. And finally, industrial practitioners and academic researchers should collaborate to ensure both that the agenda of academic research remains grounded in "real" problems, and that the results of such research is effectively and promptly converted to industrial practice.
- **Proof of concept implemen**tation. We also believe that the objective of technology transfer to industry would be greatly facilitated if the research community were to demonstrate the efficacy of their research findings via non-trivial proof-of-concept implementation projects, and make them available as open source.

Roadmap

Short-term roadmap

- Establish formal models, and work on achieving some consensus that these are the models that most accurately reflect actual concerns
- Identify critical questions (mainly conceptual) that need to be answered
- Initiate community-building activities

Medium-term roadmap

- Develop tools and implementations
- Community-building activities should have become self-sustaining – the members should consider these communities to be "natural" associations

Long-term roadmap

- Have developed tool-chains that address each aspect of system design and implementation from very high-level specification to final implementation, maintenance, and evolution. Such tool-chains should be easy for the nonexpert to use: we can declare victory when it is possible to develop large complex time-aware cyber-physical systems in a platform-independent manner (in much the same way non-time-critical systems are developed today), and have a very high degree of confidence that the resulting systems are correct.

Infrastructure for Time-Critical Systems

Introduction

With the advent of low cost COTS computing and networking elements, the advancement of distributed processing systems, such as cloud computing, has been rapid. The development of these infrastructure elements has been focused on the general case, optimizing cost and processing power, without consideration for the larger scale and critical nature of highly distributed systems. The application of generic non-deterministic infrastructure components in critical applications results in longer development, qualification/certification and overall life cycles. These critical applications are defined as cyber-physical systems (CPS). The hallmarks of a CPS are not only the close interaction between the computational and physical elements, but also the interaction of the computational elements in a distributed processing architecture. The networking infrastructure is therefore the key architectural element in a CPS.

In contrast to general-purpose computer systems, the criticality of time and time synchronization is inherent in CPS. The challenge is not to perform local or distributed computations as fast as possible, but to meet deadlines, such that the interaction with the physical environment satisfies the mission of the CPS in a deterministic way.

Hard real-time problems--where missed deadlines or incorrect synchronization of time may result in devastating consequences--are common in many CPS domains, such as automobiles, avionics, medical equipment, industrial robotics, and power plants. Networks of networked CPS hold the potential for significant critical system improvements. The current technology deployed in driver assistance safety features, such as collision avoidance, may reduce individual instances of accidents. Significant gains in the reduction or elimination of traffic fatalities, however, can only be accomplished with a time-coordinated networked infrastructure, where autonomous and semi-autonomous vehicle control can be precisely coordinated.

Similarly, improvements can be made in air traffic control, where networks of CPS onboard the aircraft and in ground control could precisely coordinate routing and queuing of departing and incoming flights. Telemedicine and telesurgery CPS suggest the ability to perform surgery on patients, where network-linked surgical teams operate in perfect synchronicity, using remote time-synchronized robotic systems. Nation-wide synchronized power grid CPS systems would be capable of monitoring load surges, line failures and source outages, and perform load balancing automatically, switching loads within a fraction of a 60 Hertz power cycle.

Such systems are very expensive to develop using the currently available technology, due to long development and test cycles. Timing in a system with networks of embedded systems is brittle; small changes in program code, compiler settings, or hardware components, may give drastic effects on run-time behavior.

An open precision timed infrastructure is an *enabling technology*; many CPS application areas can utilize the same timed infrastructure. Such open reuse of technology may vastly reduce the development and testing costs of a CPS project, as well as improving the confidence of system correctness. Reducing development cost and shorting time-to-marked gives competitive advantages, in, for instance, the automobile industry, where both cost efficiency and safety guarantees are key factors.

Standardization and *availability* of an infrastructure with ubiquitous notion of time may further lead to new innovations; new products can be developed that rely on precision, accuracy, and predictability of time, something not economically or technologically feasible without reuse of standard components

Time-coordinated CPS holds many promises. Besides enhanced operational capabilities derived from integrated devices and information systems, they allow flexible configuration and deployment, and the collection of more accurate and representative data from natural settings for longitudinal studies to support improved system performance. They also raise many challenges.

From the perspective of infrastructure technologies (other than networking technologies for connectivity), much remains to be done.

What can we do well?

Today, we are good at developing embedded systems, which interact with the physical environments using sensors and actuators. We are good at developing complex systems with soft real-time requirements, that is, systems that should, but must not meet all deadlines on time.

- Modern aircraft control systems are complex, safety critical systems of systems, transporting millions of people each year in harsh, challenging environments with excellent safety performance
- Vehicle control systems are designed and implemented with complex control systems that assist the driver with steering, braking, and traction control to maximize safety in a wide range of operating environments.
- Medical devices such as infusion pumps, patient monitoring, and support equipment such as sterilizers and surgical tables are computer controlled, reducing the dependence on human operation for life-critical medical systems.

Safety testing and system performance verification for these systems is rigorous, extensive, and has processes in place for continual improvement.

For distributed systems that make use of a common time-base for deterministic behavior, and where the nodes have stable access to the GPS system, a precise global notion of time can be established and used effectively. In a networked system, high precision and accuracy of clocks can be accomplished by using *precision time protocol (PTP)* for clock synchronization. Such systems require that the system and environment is stable; the number of nodes does not dynamically change.

Why Can't We Declare Victory?

Currently deployed cyber-physical systems are mostly stand-alone systems with proprietary designs. This is typically a result of the critical nature of these systems. Computer-assisted vehicle control, patient monitoring, and aircraft controls are designed with systems components that require deterministic operation, and as a result are closed, non distributed systems. Currently, safety-related actions in these systems are limited to computer-aided actions that are closely supervised by trained personnel because the current distributed infrastructure is a best-effort system whose real-time reliability and security cannot be ensured. The control systems aboard aircraft is computer-aided, but a functioning airport requires air traffic controllers to safely land, route, and supervise takeoff on even the most modest of airport traffic capacity.

Networked time-critical systems can also be developed, but in small scale and at high costs. The more nodes that are added to a network, the more complex applications that are deployed on the system, the harder and more expensive it is to test the system, such that it meets critical safety requirements.

The notion of time, including both predictability of execution time and measurement via clocks, is not ubiquitous. Special components and special hardware supports the notion of time, but there is no standardized way of reasoning about time.

We cannot, today, reason about time at the same abstract level as we can with functional behavior of a system. In standard system level programming languages, such as C/C++, the notion of time is a merely a performance factor, not a correctness factor. With the absence of an abstraction that includes the notion of time, software systems in CPS are not portable. New platforms, new processors, new memories, change the timing behavior of the overall system. As a consequence, such systems need to be re-tested and recertified with new (even faster) hardware.

Clock synchronization also falls short in dynamic networks (e.g. mobile and unreliable environments).

Current commercial infrastructure software assumes absolutely no liability and has many known and unknown bugs. The development of a certifiably safe infrastructure for networked systems is a long-term R&D challenge that involves not only advanced technologies but also a legally sound certification process.

Specific R&D challenges

Designing for Certification

Many devices and systems are safety critical and must be certified. At present, a heterogeneous mix of agencies performs certification for safety critical systems. Certification is desirable but needs R&D to make it possible for software and systems. Thus, it is important to develop a standards-based infrastructure of certifiable networked systems and a common set of certification criteria so that we can reduce the costs of development, approval, and the deployment of new technologies and devices.

Quality of Service

End-to-end QoS is an important concern in the operation of CPS. Quality of timing services involves three main components: precision of time, accuracy of time (compared to a reference), and predictability of time (safe and tight bounds on expected execution time or network latencies). The quality of the timing services should also be evaluated in terms of robustness, that is, how these three components are affected in an uncertain and unstable environment. The aspects of QoS are described in the following sections.

Managing Safety and Criticality

From subsystem to systems of systems, different devices and subnetworks have different levels of criticality. Data streams with different time sensitivities and criticality levels may share many resources of the hardware and software infrastructure. How to maintain safety in an integrated system is a major challenge that consists of many research issues:

• How to develop a safety interlock for the operation of interacting devices

• How to manage the flows of data streams that have different criticality on the same network

• How to mediate and manage the interactions of devices that have different criticality. How to authorize and authenticate. Who can talk to whom?

• How to support the fail-safe operation of individual devices

Security And Fault Tolerance

With respect to the propagation of time in a system, the ability to trust the time service is critical to a CPS. Systems that require hard real-time scheduling guarantees will rely both upon the assurance of data integrity, and the means to operate reliably when the data integrity systems fail.

Interoperability

Interoperability has been a major challenge in integrating systems from different manufacturers. Interoperability of system components is vital for an infrastructure to be usable in industry. A common representation and interpretation of time, its accuracy and precision, as well as the data that are created, distributed and consumed in the context of that time must be well defined. Besides careful standards, shared evaluation facilities or plug-fests (gathering of vendors and researchers) are needed, where different parties test and evaluate how systems work together.

Real-Time and Scheduling Guarantees

A complex, possibly global CPS will operate in real time with different time constraints and different sensitivities to delays and jitters. In the envisioned CPS, many types of real-time and non-real-time data traffic will share the same computing and communication resources. How to ensure the proper scheduling of real-time traffic is an important concern, and here are some of the challenges:

• What should be the policies of resource allocation and scheduling that ensure predictable end-to-end timing constraints and interoperability

• How to provide time-zone abstractions that can support monitoring and control loops that have differing time constraints, ranging from nanoseconds to hundreds of milliseconds.

Wireless Infrastructure

Wireless networking is an important enabling technology. To provide secure and reliable real-time communication, however, we face many challenges, including:

- How to improve interoperability and protect against interference
- How to improve security, reliability, and scheduling
- How to support mobility, including programming abstractions that manage mobility
- How to integrate with the wired infrastructure.

Time services

Fundamental to all aspects of Quality of Service is the establishment of the ubiquity of time in networked CPS. This universal synchronized time reference will need to be established on all levels of a complex CPS, as detailed in the following sections.

Making Time Ubiquitous on a Platform

In soft real-time applications, it is important to *measure* the time with precision, such that periodic sensing and actuation can be performed timely with little jitter. Time measurements can be used to give approximations for average-case, best-case, and worst-case execution time (WCET).

To give guarantees on safe upper bounds on WCET, however, pure measurements of time are not sufficient. Instead, static analysis on program code is necessary. The challenge is to make such methods safe (the computed bound is greater or equal to the real WCET), tight (the overestimation is small), and general (the technique is applicable to a large set of tasks).

Precise measurements of time or good methods for computing bounds of WCET are of no use if they are not available to the upper application layers. To make time ubiquitous, primitive abstractions of time must be easily available to programmers and engineers with little effort. Low level details, such as memory hierarchies, hardware threads, or voltage/frequency scaling, must not be exposed to programmers, even though it effects the execution time. As a consequence, a major research challenge is to define the right level of abstraction, including standardized interfaces that enable programmers and engineers to reason about time.

When the notation of time is ubiquitous at the system programming level, an important task is to automatically synthesize or compile the programs to a hardware platform, such that the semantics of the program and the behavior of the actual implementation on the hardware coincide. The key challenge of such translation is to preserve the timing semantics. A platform that has precise notion of time opens up for both new opportunities and challenges. Several design parameters may be considered, such as clock frequency, memory sizes, and number of hardware threads or cores.

Ubiquitous time at a platform level concerns also hardware/software tradeoffs. Tasks may be implemented in either hardware or software or a combination of both. Synthesizing the cyber part of CPS is a classic hardware/software co-design problem. To save both power and improve performance, certain parts of a system may be, for example, implemented in field-programmable gate arrays (FPGAs). In such heterogeneous design environment, the main challenges for high-confidence CPS are to guarantee the correctness, accuracy, and precision of the timing semantics at a system level.

Energy efficiency is one of the key design parameters for embedded systems. Multi-core is the current trend for improving performance without escalating the energy needs. Programming such systems, however, turn out to difficult and hard to directly apply to general-purpose programs. Multi-core systems also pose new hard challenges to predictability and precision on timing. Tradition techniques, such as caches, for hiding memory latencies and to achieving good average case performance, inherently introduces non-deterministic timing behavior, making predictability even more challenging.

On a platform, ubiquitous time is necessary not only for the processors, but all connected components, such as sensors, actuators, memory controllers, and network interfaces. Although some components with built in notion of time exists today, such as IEEE 1588 synchronized clocks, the challenge from an architecture perspective is to make all components *agree* on how to communicate with time. Interoperability and standardization are key components of making a precision timed infrastructure reality, something that require tight collaboration between standardization organizations, industry, and academia.

Making time Ubiquitous in a System

At a system level, networks of embedded systems may dynamically interact to perform overall system goals. For instance, a modern car or aircraft have many computation platforms, interacting both over wired and wireless computer networks. With a synchronized ubiquitous notion of time in such a system, new opportunities arise. Synchronous communication with near collision free traffic has the potential to changing the way time critical networks work today.

Clock synchronization, using standard such as IEEE 1588, is proven to give both precise and accurate clocks, to the level of sub-nano seconds. The current state-of-the-art clock synchronization methods assume, however, perfectly stable networks. With the trend that systems become more and more dynamic, mobile ad-hoc networks, where nodes may come and leave dynamically, pose new challenges to clock synchronization. For instance, if a node goes out of coverage for some time, the local clock of the node may drift. Such mobile systems may have an intentional instability, which is part of the intended usage. By contrast, a networked system may also be unintentionally instable, that is, nodes fail. New cross-level clock synchronization methods need to be developed that cope with such changing environment, and to provide guarantees about certain accuracy and precision within an unreliable environment.

From a systems level point of view, programming distributed systems with ubiquitous time is even more challenging than programming it at a platform level. New safe, simple, and secure methods are needed for programming different components in a distributed environment, such as sensor nodes, switches, and computation nodes. If such a low level distributed programming model also provides other time related services, such as latencies between network nodes, new opportunities for innovative application may arise. For instance, combined clock synchronization and latencies can be used for geographic position, without the need for GPS signals.

Bounds on execution time at the platform level directly relates to the clock (oscillator) at the platform. The processor' clock frequency may not be constant (in case of frequency/voltage scaling for energy efficiency), or precise and accurate due to clock drifts. As a consequence, such platform clocks do not accurately relate to synchronized real-time clock, making the relation between WCET and dynamically corrected real-time clocks non-compatible. The challenge for an infrastructure, with ubiquitous notion of time, is to seamlessly integrate execution time and synchronized distributed clocks, such that guaranteed correct programming with time is easy in a distributed environment.

Making Time Ubiquitous in Systems-of-Systems

Making time ubiquitous in systems-of-systems poses even more challenges. In such heterogeneous timed environments, interoperability and communication between different systems become difficult. Systems at different geographic locations may have different time zones, notion of precision, scales, and protocols.

One of the key challenges in a heterogeneous environment is to negotiate contracts of time quality. Different parts of the system must agree upon precision of time, accuracy of time, and bounds on response time. The latter includes both bounds on network latencies and execution (computation) time within system components. Besides technical challenges of providing such guarantees of upper bounds, ubiquitous time in systems-of-systems requires standardizations, such that different components from various vendors can function together in a heterogeneous environment.

Research Strategies and Roadmap

Near-term (5 years)

- New wired and wireless protocols that utilize synchronized time to achieve vastly improved performance for time critical applications.
- Open standards for ubiquitous time at a platform level, including low-level programming models that include the notion of time.
- New methods for time synchronization in an unreliable mobile environment.

Mid-term (10 years)

- Accepted, standardized, and proven technology for ubiquitous time at a system level.

Long-term (20 years)

- Time is ubiquitous in systems-of-systems.

Contributors

The following people have contributed with inputs to this report section:

- David Broman, UC Berkeley
- Boris Gelfand, Lockheed ATL
- Bob Iannucci, CMU
- Xenofon Koutsoukos, Vanderbilt/ISIS
- Kang Lee, NIST
- Chenyang Lu, WUSTL
- John MacKay, Progeny Systems Corporation
- Paul Miner, NASA Langley
- Miroslav Pajic, UPENN
- Eugene Song, NIST

Operating in Unstructured and Unpredictable Environments

Introduction

The operating environment is dynamic, uncertain and often unstructured in

- Transportation
- Smart Grids
- Smart Cities
- Autonomous systems
- Medical
- Aerospace and defense
  - UAVs
  - Test equipment
  - Training
    - Mixed simulated and field exercises

Functional Areas of Interest

A number of functional areas are critical for adaptable systems. These include the following.

- Adaptation and graceful degradation. In uncertain environment we cannot longer consider that a system is always in a steady state with a single goal (e.g. keep a constant temperature in a room). Instead, the uncertainty in the environment forces us to consider a wide range of situations that must be part of the system specification with an equally large variety of goals (selecting the goal according to the conditions) that the system needs to consider. As a result, it is not enough to ensure that a single goal can be achieved but we need to consider how the system evaluates new situations in the environment and dynamically changes goals. For instance, the safety of the system would need to be evaluated in the light of the evolution of the environment and how we characterize and preserve safety as this evolution occurs. This has been traditionally known as graceful degradation and evaluates how the functionality of the system diminishes as the environment conditions worsens going all the way down to a safe stop.
- Spatio-temporal. The physical perspective of CPS brings both space and time as key characteristics that must be factored in when designing a system. CPS interacting in the physical world (e.g. cars in a highway) need to be verified with respect to the time and location of their actions. It is no longer enough to ensure that a system generates the correct output value, but we also need to verify that such output happens at the correct time and in the correct location.
- Robustness. This is a special perspective of adaptability that is worth highlighting. In particular, robustness emphasizes the ability of a system to perform its function in spite of uncertainty. This uncertainty can come from the environment but also from the unpredictable execution time of modern

processors. In this case, the verification of the basic safety functionality is critical and improvements on that can be pursued opportunistically in what we call *graceful improvement*.

- Diagnostics, prognostics and forensics. As new theories and techniques for adaptable CPS are deployed in real systems it is important to collect information about their performance. Of particular importance is the information about behavior outside the operational envelope and failures. The former can be used to increase the capacity of the system to be proactive about potential accidents while the latter can be used to increase our understanding of the limitation of the current techniques to be able to improve them.
- Humans in the loop. While we expect that CPS would be able to replace humans in some functions, CPS may have a more significant role augmenting human capabilities. In this situation, the human cognition bandwidth would play a critical role in the innovations for CPS. In particular, it would be necessary to provide the human operator with the proper amount of information to operate the system at the proper pace. For instance, in order to enable a car with an autonomous convoy mode, i.e., where a car drives itself following the car in front of it, the transfer from autonomous to manual driving needs to consider the limits of the reaction time of the human driver.

Benefits of good solutions to the above for cyber-physical systems will pay off in infrastructure in which we can put greater trust, improving our social, economic, and military security.

Failure to develop a sound scientific and engineering approach to dealing with these factors will result in brittle cyber-physical systems that are likely to fail in disastrous ways.

What can we do well?

We are currently capable of building CPS with some degree of adaptability. However, this capacity is limited to the following characteristics.

- We are able to build reliable CPS, so long as the <u>system and its environment are well structured and static.</u> However, problems grow as the environment becomes more dynamic and unpredictable, as systems grow by composition, evolution, and accretion (e.g., open vs. closed systems).
- We can provide <u>reliability and tolerance of predictable faults, using redundant HW/SW with complete duplication</u>. In other words, we have effective techniques when it is possible to enumerate all potential failures building replication of complete subsystems. However, this is costly, and does not scale well.
- We can build <u>systems with well-defined static workloads</u>, and guarantee behavior with respect to time. However, when the workload is not static our current ability to adapt to this variation is limited.

- Timing is manageable (*with difficulty*) within closed network systems, e.g., a closed LAN. Once communication is more open, such as in an ad-hoc network, time gets out of hands.

What can we *not* do well?

Our knowledge to build adaptable systems has multiple limitations. These limitations include:

- Modeling of uncertainty. We do not have effective ways to model uncertainty in a way that is useful to inform the analysis and design decisions of CPS. In particular, a model of the uncertainty of the environment should enable us to determine the bounds of the behavior of the system around this uncertainty.
- Current model of time is inefficient/ineffective to coordinate physical and cyber processes in uncertain and unstructured environments (wireless, mobile, open roads, etc). This is due to the fact that current coordination techniques rely on synchronization techniques that, while precise, are not robust enough for these environments. There are no techniques for graceful degradation or graceful improvement of time references or the combination of multiple sources of time references.
- We cannot prove the properties of systems that learn. Learning techniques are increasingly important in every system from voice recognition, to searches in the internet, to autonomous driving. However, because the behavior of the system changes as it learns, current verification techniques are not able to take these changes into account.
- We cannot make trade-offs between multiple quality metrics such as performance and robustness or mission thoroughness and efficiency. Our current techniques focus on the optimization of one metric ignoring the others. Such an approach rarely matches the expectation on real systems. This is especially true in uncertain environments.
- Uncertain environment creates highly dynamic workloads, difficult to predict or model, so scheduling becomes difficult. In particular, we do not know how to guarantee correct timing of systems with dynamic workloads, including overloads.
- Design systems that maintain safety and performance under timing uncertainty and timing variability. Current approaches of model-based design of distributed systems largely assume the underlying hardware platforms, operating systems and middleware infrastructure provide high-fidelity and constant precision timing. As these assumptions are overly constraining in the development of systems where the underlying substrates are unreliable (e.g. wireless channels, heterogeneous COTS architectures, environmental effects on system operation, etc.), new modeling approaches based on timing theory which incorporate uncertainty and variability in timing and precision are required to guarantee safety and performance during derangements of synchronization across the distributed system.

R&D Challenges

The current limitations of the state-of-the-art confront us with the following challenges.

- CPS require a concept of time to allow them to synchronize with the physical world. Hence, it is necessary to develop techniques that can preserve time predictability in the presence of dynamic changes in workload, e.g. due to adverse conditions in the environment.
- The increased uncertainty that CPS face requires models of systems that embody uncertainty and unpredictability in both the environment and the behavior of system components (including learning or adaptive algorithms) that still allow us to provide guarantees. The uncertainty model and related technology must be able to cover:
    - The modeling of failing parts, as well as the fault-containment mechanisms that isolate these failures
    - Analysis techniques that verifies the behavior of the system in the presence of failures
    - The modeling of system assumptions that bound the verified behavior of the system as well as techniques to adapt and contain the potentially adverse effects when these assumptions are violated.
    - Techniques to combine self-stabilization properties in systems with fault tolerance
    - Techniques to preserve coordination in the presence of communication failure / attacks
- CPS operating in an uncertain environment cannot be characterized by a single metric. As a result, we need to develop techniques to achieve safety, robustness, etc. without loss of efficiency or, rather, we need to develop techniques to achieve a good trade-off, or navigate trade-off space. This means that we need to avoid the optimization of a single property (e.g. resource consumption, performance) and instead create a better description of the property interactions. For instance, how performance and robustness interact.
- Given the dependency of CPS on synchronization mechanisms it is critical to develop new techniques to provide a synchronization infrastructure that is resilience to failures, attacks, or disasters (hurricanes)
- CPS working in uncertain environment will be faced with failures of components and varying conditions in the physical world outside their normal operation envelope. As a result, we need a framework for graceful degradation in order to allow us to build and analyze systems with predicable degradation of operation in the presence of these failures.
- Complementary to graceful degradation, we need a new characterization of required robustness (i.e. criticality) of the different functionality of the system with mixed-criticality requirements to preserve safety or to improve the performance of the system. This includes new metrics that take into account these different levels of criticality, new mechanisms that adapt to uncertainty while honoring this robustness needs, and models to analyze

mixed-criticality systems compatible with certification models such as the layered certification requirements of DO178B

- The implementation of CPS for uncertain environment requires a robust diagnosing infrastructure. At the same time the architecture (both hardware and software) of the CPS must be built to enable this diagnosability. This function should be used as a self-diagnosing capability embedded in their logic enabling them to be proactive to deviations of the normal operation. Similarly, it should enable the construction of audit trails to support certification, accident investigation, and learning. The diagnosing infrastructure must be able to take advantage of tightly synchronized clocks to correlate events in a large number of distributed processors.
- We need to develop techniques to incorporate timing of different levels of precision and availability. In particular, functionality that must be highly available and critical should not rely on timing mechanisms that are not highly available. At the same time it would be important to take advantage of high precision clocks whenever they are available to improve the performance of the system. This necessitates the combination of clocks from different sources (e.g. GPS, quartz clocks, AC cycles) in a consistent manner. Similarly, it would be critical to manage trust in time according to the origin of the clocks.

Research Strategies and Roadmap

3-year roadmap

In the short term we will be able to reach some useful initial milestones including:

- Developed synchronization protocols for at least two time sources.
- Developed techniques for self-diagnostics for small scale CPS.
- Developed a modeling framework to describe systems with more than one alternative goal that are selected according to the conditions of the environment.

5-year roadmap

In the medium term we expect that the research efforts in this area would reach the following milestones.

- A sound framework to design and analyze systems with graceful degradation under component and timing failures.
- A well-characterized model that captures and bounds the uncertainty for at least some environments, which in turn is used to define and determine the behavior of a CPS system in the presence of such uncertainty.
- Robust models of coordination with multiple timing sources that increases the robustness of the coordination while improving its performance whenever high precision timing sources are available.
- Effective techniques to provide timing guarantees in large-scale systems such as the Internet.

- Provable analytic models to integrate the optimization / tradeoff of both time predictability and robustness for systems operating in uncertain environments (for single criticality systems).
- Real-world prototypes that support CPS with mixed-criticality requirements.
- Real-time techniques for performing self-diagnostics in distributed CPS.

10-year roadmap

In the long term we expect to achieve the following milestones.

- Fielded CPS distributed systems with multiple timing sources with a robustness larger than the robustness of any single timing source.
- Built systems with verified bounded behavior with respect to its model of uncertainty.
- Developed analytic models that tradeoffs across multiple trade-spaces. For instance, robustness performance, mission thoroughness vs. efficiency, pollution vs. economy, etc.
- Built systems that can tolerate large number of failures types
- Built mixed-criticality systems with multiple tradeoffs spaces
- Built distributed CPS that can operate at different levels of connectivity and synchronization continuously

Unsorted points from open discussion:

- Need robustness wrt inaccuracies/gaps in models
  - John Rushby's epistemological risks
  - New elements added to environment
  - New compositions of systems
  - Changes to systems
  - Violations of models/specifications, or models that don't fit reality
- Approaches
  - Bound the unknowns, and then bound the behavior of the system
  - Need to accept that accidents will still happen
- Need high fidelity models of the operating environment, including physical models of cyber components (e.g., antenna as moving object, leaves change to yellow from green)
- Need robustness w.r.t inaccuracies/gaps in models
  - John Rushby's epistemological risks
- Assume things go bad and still try to make guarantees
- Bound the unknowns, and then bound the behavior of the system
- Accidents will still happen
- What are the benefits to people, that we are trying to achieve?
- Maintain safety while advancing throughput, and reducing fuel usage (and hence CO2 pollution)
- All CPS are crucially dependent on correct timing

Verification, Validation and Certification & Trustworthiness and Security

Draft by John, Joe, and Karen

What Can We Do Well?

The present state of the art is fairly effective at building individual cyber-physical systems, even those that are safety-critical, such as medical devices, anti-lock car braking systems, aircraft flight control, and so on. There is certainly progress to be made in reducing costs, increasing reliability and fault-tolerance, and quantifying confidence in certification, but the present state of the art is tolerable.

What Can We Not Do Well?

What we cannot do well is connect individual systems together as components so that they function as a coherent and safe integrated system, in a compositional manner. By compositional, we mean a modular approach in which the properties of the integrated system follow from those claimed for the component systems, without concern for what is inside those components, nor how they are built.

Failures of integrated systems, such as recent power and telephone blackouts and aircraft incidents are invariably traced to some unexpected interaction among components that can only be understood by looking inside them and at the ways they operate and interact: compositionality (or modularity, if you prefer) has broken down.

Synchronized time is a powerful organizing principle for compositional integration of systems. In tightly integrated systems, such as the suspension, steering, brakes, and powertrain of a car, synchronization allows these component subsystems to operate to a common "heartbeat" so that they share a common view of each others' state and do not clash over access to shared resources such as communication buses.

In less tightly integrated systems such as those that interact over the Internet, where there are no guarantees on how quickly information can be exchanged, timestamps generated by synchronized clocks allow separate components to achieve a common view of the order in which events occur and a consistent view of the state of the overall system.

Synchronized precision time provides new opportunities: for example, electric power transmission can be better managed when the phase is accurately known at remote locations.

In systems that are not well integrated, reliable timestamps provide a critical resource for forensic reconstruction of events leading to failure or compromise.

Until recently, the benefits of synchronized operation or of reliable timestamps could only be achieved in specially-engineered and somewhat costly systems that guarantee the maximum time taken to exchange information between nonfaulty components, even when other components are faulty (and might therefore be "babbling" on the communication buses). Examples include the SAFEbus, AFDX, and

TTE architectures used in commercial aircraft. Recent advances, such as hardware support for protocols such as IEEE 1588, highly accurate local clocks (e.g., chip-scale atomic clocks), and GPS bring many of the benefits of synchronization to more general classes of systems.

However, these benefits are not guaranteed because most of the recent advances were not engineered for adversarial environments, they do not inherently have the reliability and fault tolerance required for safety-critical applications, and they cannot fully compensate for other vulnerabilities in general-purpose systems.

The big opportunity for research in this area is to extend the benefits conferred by recent advances through development of methods that ensure synchronization in adversarial and faulty environments, that can provide trustworthy timestamps, and that provide frameworks for the construction and certification of safe and reliable systems based on these.

R&D Challenges

At present, it is impossible to fully trust a timestamp. Even if the timestamp is signed by a trusted authority, we have no assurance beyond the reputation of the authority that its own clock is reliable. Even if the authority claims to be synchronized to GPS, the assurance is not strong, because GPS can be spoofed. The research challenge is to develop a method for trustworthy timestamps.

Since GPS is easy to block and to spoof, methods for providing more assured methods for access to time and position information, perhaps by augmenting WAAS, should be investigated.

Similarly, IEEE 1558 is not strongly fault tolerant and research is needed into methods for augmenting the security, fault tolerance, and resilience of synchronization and time distribution methods.

GPS and other precision timing methods allow the construction of synchronized systems that are not synchronous (i.e., there is no guarantee on the time required for message exchange between non-faulty components). This does not correspond to any standard model for partial synchrony in distributed systems and research is needed to characterize its properties, effective ways to exploit it, and convenient methods for programming on top of it.

Building on the above, research is needed into methods for exploiting synchronization to support fault tolerance, security and resilience, and the provision of trustworthy services. We note that time can also be used as a channel for covert communication and research is needed to detect, block, and ensure the absence of this channel.

And building on all these, research is needed into methods for providing credible assurance and certification for timing services and systems the depend on these.

Societal Impact

Society is massively dependent on CPS at all levels, from individuals through the enterprise to national infrastructure. Very few of these systems are able to resist a moderately skilled and determined cyber-attack. Examples range from the complete takeover by a remote attacker of the brakes, throttle, and steering of a domestic automobile in motion (CarShark), the downing of an RQ-170 sentinel in Iran, and the Stuxnet worm.

Time services are particularly vulnerable, and likely to become a point of attack as others become better protected. At present, there is no source of precision time that cannot be disrupted or spoofed: it is impossible to trust a timestamp. This not only provides opportunities for attack, it prevents trustworthy reconstruction of timelines for forensic investigation of accidentally or deliberately induced failures. Fully trustworthy provenance is likewise impossible the sequence of evidence, ideas, and materials in scientific, legal, and industrial development chains.

On the other hand, ubiquitous availability of trustworthy precision time would be an enabler for innovation and development at all levels, supporting not only new products and services, but increased reliability and trustworthiness in existing systems, and reduced costs.

Examples include integration and "closing the loop" for multiple medical devices serving the same patient, integration of new and green energy sources into the power grid. Synchronization also facilitates greatly increased utilization of existing infrastructure, such as roads, air, power, and enables energy efficient monitoring of large structures, machines, and massive and remote industrial plants (such as oil pipelines).

Furthermore, the massive costs of remediation for security vulnerabilities and general unreliability of current large-scale systems would be greatly be reduced if more trustworthy system architectures and methods for integrating systems into larger systems of systems were developed. Synchronized precision time has the potential to be a "game changer" in this regard, since it assists coherent system-wide knowledge of the system state.

Economic Impact

Some of this is implicit in the societal impact.

Other advanced nations are making large investments in research and coordination in this area (see, e.g., http://www.artist-embedded.org/artist/), and industrial competitors are pursuing large development efforts. Certain classes of tools for timing analysis are available only from foreign suppliers (see, e.g., http://www.absint.com/).

The competitiveness of US products will be greatly enhanced if trustworthiness and security can be transformed from liabilities and vulnerabilities into positive assets, and the New Clockwork has the potential to be a "game changer" in this respect.

The underlying mechanisms of the New Clockwork (GPS, IEEE 1588, chip-scale atomic clocks etc.) will be available to all. The singular opportunity for the USA is early recognition of the system-level opportunities created by these, and development--spearheaded by effective investment in R&D -- of technology and know-how that leverages these capabilities into an ability to construct large-scale systems of systems in a modular or compositional manner that are dependable, trustworthy, and affordable.

Appendix. Workshop Attendees

Abdelzaher, Tarek, UIUC

Adam, Nabil, DHS

Amelot, Julien, NIST

Anderson, James, The University of North Carolina at Chapel Hill

Arnold, Doug, Symmetricom

Baker, Theodore, National Science Foundation

Baruah, Sanjoy, UNC

Branicky, Michael, Case Western Reserve University

Broman, David, UC Berkeley

Brooks, Christopher, UC Berkeley

Chandhoke, Sundeep, National Instruments

Chopra, Nikhil, University of Maryland

Dailey, Daniel, FHWA/UW

de Niz, Dionisio, Carnegie Mellon University - SEI

Dey, Mary, Vanderbilt University

Doboli , Alex, State University of New York at Stony Brook

Dyson, Anne, Vanderbilt University

Eidson, John, UC Berkeley

Fainekos, Georgios, ASI

Gelfand, Boris, Lockheed Martin

Gilkes, Shannon, OASD(R&E)

Gill, Helen, NSF

Goldman, Julian, Mass General Hospital/Partners HealthCare

Grijalva, Santiago, Georgia Tech

Gupta, Rajesh, UC San Diego

Hochschild, Peter, Google

Iannucci, Bob, Carnegie Mellon University

Januszewski, Joseph

Jones, Paul, FDA

King, Frankie, Vanderbilt University-ISIS

Koutsoukos, Xenofon, Vanderbilt University

Kuehn, David, DOT

Kumar, Ratnesh, Iowa State University

Lee, Kang, National Institute of Standards and Technology

Lee, Insup, University of Pennsylvania

Lee, Edward, UC Berkeley

Lenzen, Christoph, The Hebrew University of Jerusalem

Li-Baboud, Ya-Shian, NIST

Lu, Chenyang, Washington University

Lucier, Ernest, NCO/NITRD

Luker, Mark, National Coordination Office

MacKay, John, Progeny Systems

Malekpour, Mahyer, NASA

Maroti, Miklos, Vanderbilt University

Martin, William, National Security Agency

May, Michael, Dept. of Defense

McNamee, Dylan, Galois, Inc

Miner, Paul, NASA Langley Research Center

O'Donoghue, Karen, ISOC

Pajic, Miroslav, University of Pennsylvania

Phan, Linh Thi Xuan, University of Pennsylvania

Rajkumar, Raj, Carnegie Mellon University

Rushby, John, SRI

Song, Yuyin (Eugene), NIST

Strawn, George, NCO/NITRD

Tabuada, Paulo, UCLA

Tilbury, Dawn, University of Michigan

Wavering, Albert, NIST

Wesson, Kyle, The University of Texas at Austin

Zellweger, Andres, JPDO