Open Challenges in Real Time Embedded Systems

Lui Sha, CS, UIUC
May 3, 2004

lrs@cs.uiuc.edu

1.0 Introduction

> "*FAA cancelled the AAS program, casting aside 11 years of development time and, according to GAO, wasting more than $1.5 billion of taxpayer money*." [1]

> "*F/A-22 problems have limited DOD's ability to upgrade its aging tactical aircraft fleet. If the F/A-22 program had met its original goals, the Air Force could have been replacing older aircraft with F/A-22 aircraft over 7 years ago. Now, however, it will not begin replacing aircraft until late 2005 at the earliest*." [2]

Serious problems in developing large and complex real time systems are not isolated incidents. Rather, they are now the rule rather than exceptions. These problems are the manifestation of building large systems with a complexity higher than what can be handled by existing technological infrastructure. They are the reflection of under-investment of R&D in this critical area.

From 80's to now, we have witnessed the changing trends in real time mission critical systems. First, the open system movement became established. Real time mission critical systems evolved from vertically integrated systems towards horizontally integrated systems in which multiple vendors supply components based on open standards or de facto commercial standards set by industry leaders. Technologically, the system architecture has changed from federated system architecture to integrated system architecture during the 90's, and then to the current new generation of system of systems. Each shift has created enormous challenges to the available technological infrastructure.

2.0 Challenges from Architecture Paradigm Shifts

Under federated system architectures, a system is characterized by a collection of private hardware resources dedicated to a special mission capability, a small number of high volume and high variability sensor data streams on dedicated links, loosely coupled distributed actions, and hardware based isolation and protections that are the results of private hardware resources and dedicated communication links. The existing technological infrastructure for real time mission critical systems was mainly developed under Office of Naval Research's Real Time Systems Initiative during the 80's. In fact, the sample challenge problems to motivate the research were taken from federated systems, where the common assumptions are 1) hard real time periodic messages are dominant tasks; 2) task sets are relatively static. Online change of task sets (mode changes) is infrequent and synchronized; 3) the worst-case execution times and average execution times for periodic and aperiodic tasks are known or can be estimated accurately.

2.1 Challenges from Integrated System Architectures

In the 90's, system architectures started shifting from federated systems architecture to integrated system architectures, which is characterized by extensive resource sharing ranging from sensors, processors to communication channels. Instead of a small number of high volume and high variability data flowing on dedicated links, we now have a large number of high volume high

---

[1] http://www.house.gov/transportation/press/press2001/release15.html

[2] http://armedservices.house.gov/openingstatementsandpressreleases/108thcongress/03-04-02li.pdf

variability sensor data streams on shared channels, and the distributed actions become tightly coordinated. The extensive sharing and tight integration has stretched the existing real time resource management theory and tools to the limit.

In the integrated systems, the large number of shared processors and networking channels allows a very large number of potential configuration options at the design time and a large number of system reconfiguration options at runtime. The existing infrastructure can answer if a particular configuration is schedulable but offers little help to the architectural system decomposition and system configurations, at which time there is significant uncertainty about the task set parameters. Integrated system architecture demands resource management technologies evolve from answering schedulability question to technologies and tools to support system architecture and configuration optimization under uncertainties in the task set parameters. It should be noted that this is not a simple task like handling parameter uncertainties in linear programming, because of the discrete constraints in the widely used static priority schedulability analysis. To date, we still do not have a suite of technological tools to support the optimized use of resources during the system design phase for integrated system architectures.

In addition, the hardware based isolation and protection under federated architectures has been mostly replaced by software based isolation and protection, which could, unfortunately, be compromised by the all too frequent software bugs. The integration of real time and fault tolerance, especially software fault tolerance, became vital when resources are extensively shared but poorly protected. Optimized resource management must take into the account of stability. That is, the essential service must be delivered reliably and cannot be compromised by the faults and failures from useful but non-essential features. We are still a long way off from a matured technological infrastructure that provides us with dependable and optimized real time resource management during the design phase.

2.2 Challenges from System of Systems

Although we yet to have a matured technological infrastructure to support integrated system architectures, we are now developing complex system of systems. Some of the foundations of real time resource management developed under federated system architectures are not only stretched but in fact called into questions. In a system of systems, each system is autonomous to others. The assumption of a relative static task set that changes infrequently and synchronously becomes questionable. To reason end-to-end timing delays in a system of systems, we must view it as a collection of systems with a set of distributed *soft coordination states*. That is, in spite of the coordination settings during runtime, each system could change its own state asynchronously in reaction to severe events arising from its local environment. To handle the uncertain and variability, we need to shift the paradigm of real time resource management from an open loop approach, where we can know sufficient details ahead of time and plan out everything in advance, to a feedback approach designed to deal with uncertainties. It is encouraging to see the emerging interests from both computing and control communities on the feedback control based approach to handle the performance engineering challenges[3]. However, much still needs to be done in this area.

A system of systems is often a large distributed system, where keeping distributed views and actions timely and consistently is at the heart of collaborative actions. Ideally, we would like to keep distributed views, state transitions and actions consistency with each others. In business systems, the consistency of a distributed system is managed by atomic operations. Simply put, atomic operations wait for every working component

[3] K.E. Årzén, A. Cervin, J. Eker, and L. Sha: "An introduction to control and real-time scheduling co-design.'' In CDC 2000, Sydney, Australia, December 2000.

ready and then commit the operations. However, this may not be viable for real time systems. The train must leave the station without waiting for everyone getting onboard, so to speak. However, those components that are left behind with outdated views and states must quickly resynchronize itself with the system in a relatively short window of time. If more and more components are out of synchronization, the distributed system would fail. How to handle the interactions between real time, consistency, divergence and timely synchronization between distributed views, states and actions is a serious challenge. As networked embedded system of systems grows larger and the coordination becomes tighter, so will be the impact of this technological challenge.

Another characteristic of a system of systems is that a variety of real time, fault tolerance and security protocols are used in different systems, because most of systems of systems are integrated, not built from scratch. One area that has particularly rapid changes is security protocols because a system of systems often faces cyber attacks from many places at once. Most of these security technologies have been, however, developed in the context of general purpose computing and networking without stringent real time and dependability requirements. It is well known that perfectly fine medicines when taking alone can react badly when taking together. From time to time, technologies developed separately can react badly when used together. For example, the well-known priority inversion problem in many systems on earth also nearly doomed the Mars pathfinder mission[4]. Pathological cross-domain technology interaction is a serious potential threat with wide ranging implications. However, this is not an easy problem to solve because the scope of modern technologies is so large and complex. To advance any area, one must specialize. As a result, we have specialized real time, fault tolerance, security, communication and control communities focusing on improving the results in one dimension with little attention on how separately developed protocols may interact. We need to create a forum for the co-development/integration of real time, fault tolerant, security, communication and control protocols. Research is needed to formally verify that protocols do not invalidate each others' pre-conditions when they interact.

3.0 Summary and Conclusion

Technologically, the real time embedded system architecture has evolved from federated system architecture to integrated system architecture and then to system of systems. Each shift has brought about enormous challenges to the available technological infrastructures. The current generation of technological infrastructure is mostly based on 80's technologies coming from ONR's Real Time System Initiative. It is inadequate with respect to 90's integrated system architecture in the sense that the infrastructure does not provide designers with the tools to optimize the resource utilization and create dependable real time system architecture. We are now building complex system of systems. The gap between the technologies that are needed and what is available further widened. There are many open problems that need to be addressed.

However, in spite of the significance of real time mission critical systems, the Federal investment in this area is woefully inadequate. Apart from the ONR Real Time System Initiative in the 80's, there has not been a focused significant investment in this area for a very long period of time. Large real time mission critical systems has become dependent on commercial products without the necessary technologies to support the combined and stringent real time, fault tolerance and security requirements. At this point, serious quality and budget overrun problems in large real time mission critical systems have become the rule rather than exception. This is unacceptable.

[4] http://catless.ncl.ac.uk/Risks/19.49.html#subj1