

CHAPTER 1

THE SEC VISION

HELEN GILL and JOHN BAY

1.1. THE LEGACY OF CONTROL TECHNIQUES

At the time of conception of the DARPA Software Enabled Control (SEC) program, control research was proceeding down a path determined by old views of the computational and systems context. The assumptions were as follows: highly constrained sensing and actuation, limited processing and communication resources, computational intractability of large or even moderate state spaces, poorly characterized and unpredictable switching effects, and target systems that operated independently and without interaction with other systems.

Control theory and engineering have a remarkably successful history of enabling automation, and information-centric control is by now pervasive. Yet today's controllers are conservative: Being products of overdesign, they often yield underperformance. Their designs are statically optimized for nominal performance, around simplified time-invariant models of system dynamics and a well-defined operational environment. They also fail in unexpected circumstances: control vulnerabilities that arise in extreme environments are frequently ignored. System modification (reconfiguration, damage, failure) may demand large changes in the controller, perhaps online during operation.

Research in adaptive control has sought to accommodate change through the use of online feedback to the governing parameters; robust control research introduced periodic recalculation of the controller, using model-based prediction. Relative to simple PID and set-point control, these model-centric strategies yield improvement. However, there are several problems. Disruptive events occur unexpectedly, not periodically, and the changes required may be dramatic. Principles and support are lacking for reactive but systematic online reconfiguration of models and software. A popular modern

strategy is to enlarge models for “full-envelope design,” attempting to accommodate an ever-larger span of configurations and environmental conditions in a single control law. This results in monolithic, flattened models with immense state spaces.

1.2. THE LEGACY OF CONTROL SOFTWARE

Mirroring the progress of control engineering, the emerging reality in information technology was one of great promise: exponential growth in processing speeds; new and better-used communications modalities; new storage capacity for capture and online exploitation of information (both sensed and model-generated); and new strategies in software composition that enabled extensible, configurable open software and systems. We had developed device networks, smart sensors, programmable actuators, and systems-on-a-chip, along with distributed objects, real-time operating systems and code generators, and application-specific design tools, but the synthesis of embedded control with these tools was a new problem.

The challenge for the SEC program was how to exploit software and computation to achieve new control capabilities. The program was formulated to create the necessary linkage between physical systems and the software and computation strategies needed to enable next-generation control systems. It was also intended to jump-start new technologies with updated assumptions about distributed embedded systems.

1.3. A NEW PERSPECTIVE ON SOFTWARE AND CONTROL

With the advent of networked sensors and actuators, distributed computing algorithms, and hybrid control, the term “system dynamics” has taken on a whole new meaning. Whereas it used to bring to mind only ordinary differential equations with perhaps some parameter uncertainty, noise, or disturbances, we can now include dynamic tasking, sensor and actuator reconfiguration, fault detection and isolation, and structural changes in plant model and dimensionality. Consequently, the ideas of system identification, estimation, and adaptation must be reconsidered.

This new perspective of the world also requires new models for control software implementations. But we must avoid the temptation to think of software as simply the language of the implementation. Control code—particularly embedded control code—is a dynamic system. It has an internal state, responds to inputs, and produces outputs. It has time scales, transients, and saturation points. It can also be adaptive and distributed. As any control engineer knows, if we take this software dynamic system and couple it to the plant dynamics through the sensor and actuator dynamics, we have a composite system whose properties cannot be decided from the subsystems in isolation.

Thus, when we put an embedded controller on a hardware platform, we have not only a coupled system with significant off-diagonal terms, but a distributed hybrid one at that. To borrow from the computer engineering lexicon, we have a problem in control/software co-design. The control design is evolving through the development of hybrid optimal control, reachability analysis, multiple-model systems, and parameter varying control. The software is being facilitated by distributed computing and messaging services, distributed object models, real-time operating systems, and fault detection algorithms. What we seek in the SEC program is a mutual catalysis of such control and embedded software technologies that will push the boundaries of performance, complexity, and applicability.

1.4. SOFTWARE ENABLED CONTROL FOCUS AREAS

To address these new challenges in software control, four focus areas were originally identified, with a fifth added after the first year's progress.

1.4.1. Managed Models for Predictive Control

Parallel distributed processes, together with time management strategies available in modern real-time operating systems, are particularly effective in facilitating multiple model control approaches. By maintaining system identifiers and parameter estimators in separate processes, more diverse plant models can be generated without the immense time penalties incurred if the estimators share a CPU with the controller. Such models can include previously time-prohibitive components such as nonlinear filters, disturbance predictors, neural network training, and environment dynamics. If the transitions between these models can be properly managed through hybrid controllers, then there are few limits on the number or complexity of plant models and controller configurations that can be used.

1.4.2. Online Control Customization

As these models converge, though, we require a service for identifying the appropriate one, switching it online, and managing this transition in time. The result will be that real-time control will no longer be limited to single-model methods, and we will see improvements over gain-scheduled approaches that assume a fixed controller structure. This represents a revolution in adaptive control, from parameter or gain adaptation in slowly time-varying systems to major structural changes or varying plant dimensionalities.

In addition, the use of smart sensors and programmable actuators adds flexibility to the control configuration and presents numerous options for fault detection, isolation, and reconfiguration. Because these possibilities change the controller's input and output mapping, they also represent control modes and must be managed accordingly. Of course, the challenge in

implementing such diverse control configurations in a single system is transition management, both logical and temporal.

1.4.3. Hierarchical, Multimodel, and Multimodal Coordinated Control

The distributed controllers enabled by emerging real-time middleware support can consist of hierarchical systems, integrated subsystems, or independent confederated systems, such as multivehicle systems. All these require coordinated control. Currently, coordination is managed in ad hoc supervisory levels of software control, usually with inadequate vertical integration among hierarchical levels. Today's prototypical control system is one that has one or only a few modes of operation and is implemented (often at the expense of significant manual effort) as a strict and static hierarchy of supervisory levels. These controllers are realized in software as nests of loops. If they coordinate, the strategy is "hard-wired." Such monolithic software is difficult or impossible to change. By contrast, today's trends suggest that control systems increasingly are needed that are much more dynamically configured; that can manage extremely large state spaces; that have open, modular designs; and that enforce a more precise relationship between low-level physical control and supervisory or coordination levels.

1.4.4. Open Control Software Tools and Services

Software services for implementing embedded controllers have remained at a primitive level and have not addressed the embedded software needs of control systems designers. Attention typically has focused on low-level real-time operating system services that are not at a level of abstraction well matched to control software design. These services only poorly support such techniques as adaptive and robust control and hybrid mode transition, all of which may require coherent dynamic reconfiguration of processes and data streams.

Techniques are emerging for open distributed systems that address these needs. They are offered as service layers in real-time operating systems and as portability and interoperability services such as the real-time common object broker, RT CORBA. However, they are not yet exploited in control systems. The monolithic code currently synthesized by control design frameworks lacks suitable levels of abstraction and is neither well-tuned nor amenable to improvement by hand. It is neither portable nor analyzable for such problems as software and communication transients and interference with other critical functionality in the system. Current technology for reasoning and synthesis does not sufficiently address the many interacting issues: the composite of continuous and discrete behavior, software and control, offline design, and online operation.

1.4.5. High-Confidence Systems and Software

Pushing the boundaries of performance and complexity in functionality means that there will be commensurate problems with validation, verification (V&V), and certification for each application. Automation does us little good if we have no confidence in the tools we have created. However, quality assurance and certification analyses generally lag the functional technologies and are too often measured by after-the-fact performance metrics rather than being treated as design-time constraints. An emerging focus area for the DARPA SEC program is the treatment of V&V and certification as design-time criteria, as well as to develop scalable checking and proof technologies that will ultimately result in correct-by-construction or “autocertifiable” code. Although such constraints are application-specific, they are as real as other timing or quality of service constraints. Yet, they have not been addressed in control software.

1.5. THE DARPA SOFTWARE ENABLED CONTROL PROGRAM

It was against this context that the Software Enabled Control program was formulated. The program objective was to unite computer science, systems software, and control technologies, with a primary goal of improving actual control performance in the face of demanding environments and requirements for interactive, cooperative control. However, instead of attacking the problem only at the higher levels of supervisory control, which typically exploit only discrete methods, it was determined that even low-level differential control of reasonably uncomplicated systems could and should be targeted. Thus, the program challenges were seen to lie in three areas: (1) so-called “simple systems,” (2) systems built by integrating subsystems at the time of design or configuration, and (3) multisystems, which might form dynamically yet have tight physical constraints on interoperation. Examples include: fuel conservation in a single vehicle; coordinated braking, fuel management, and transmission control within a vehicle; highly automated management of “swarms” of autonomous air vehicles; redundantly instrumented and actuated systems capable of reconfiguration under catastrophic failure; resolution determination for a group of aircraft in a sudden collision avoidance situation or other joint encounter; or coordination of varying, proximity-based enclaves of safe, energy-optimizing vehicles in an intelligent highway system or on a battlefield.

In the pursuit of these goals, the SEC program was started in late fiscal year (FY) 1999 and was fully underway in FY 2000. It focused initially on four technical areas: the active management of state models and online information for predictive control; hierarchical, multimodal coordinated control; online control redesign and software reconfiguration; and software technologies for open control systems. In FY 2001, a fifth emphasis was

added in high-confidence software and systems, to build on and guide the technologies developed under the initial four areas. Building on the research begun in this DARPA program, a new base program in Embedded and Hybrid Systems has been created at the National Science Foundation to support and sustain the ongoing effort, both foundational and experimental, that will be required to realize the SEC vision.

As seen in this volume, the vision of software-enabled control is starting to bear fruit: New architectures, algorithms, and models are emerging. At the same time, new directions and opportunities continue to emerge, and the vision appears likely to outlive the program itself!

ACKNOWLEDGMENTS

The superb technical leadership we received from Dr. David Tennenhouse and Dr. Shankar Sastry, in their service as Directors of the DARPA Information Technology Office, has earned our lasting respect and gratitude. They provided wisdom and thoughtful guidance, wit and encouragement, and (perhaps most important) an enthusiastic commitment to the merit of the SEC endeavor. Reed Morgan (now retired), Bill Koenig, and Ray Bortner of the USAF AFRL have been invaluable partners. We owe permanent debt to Jessica Greenhalgh, Nikki Morris, and Carmen Whitson for their cheerful, professional, and dedicated efforts to ensure the smooth operation of the SEC program. We would also like to thank Mark Swinson and Janos Sztipanovits for their encouragement and support.