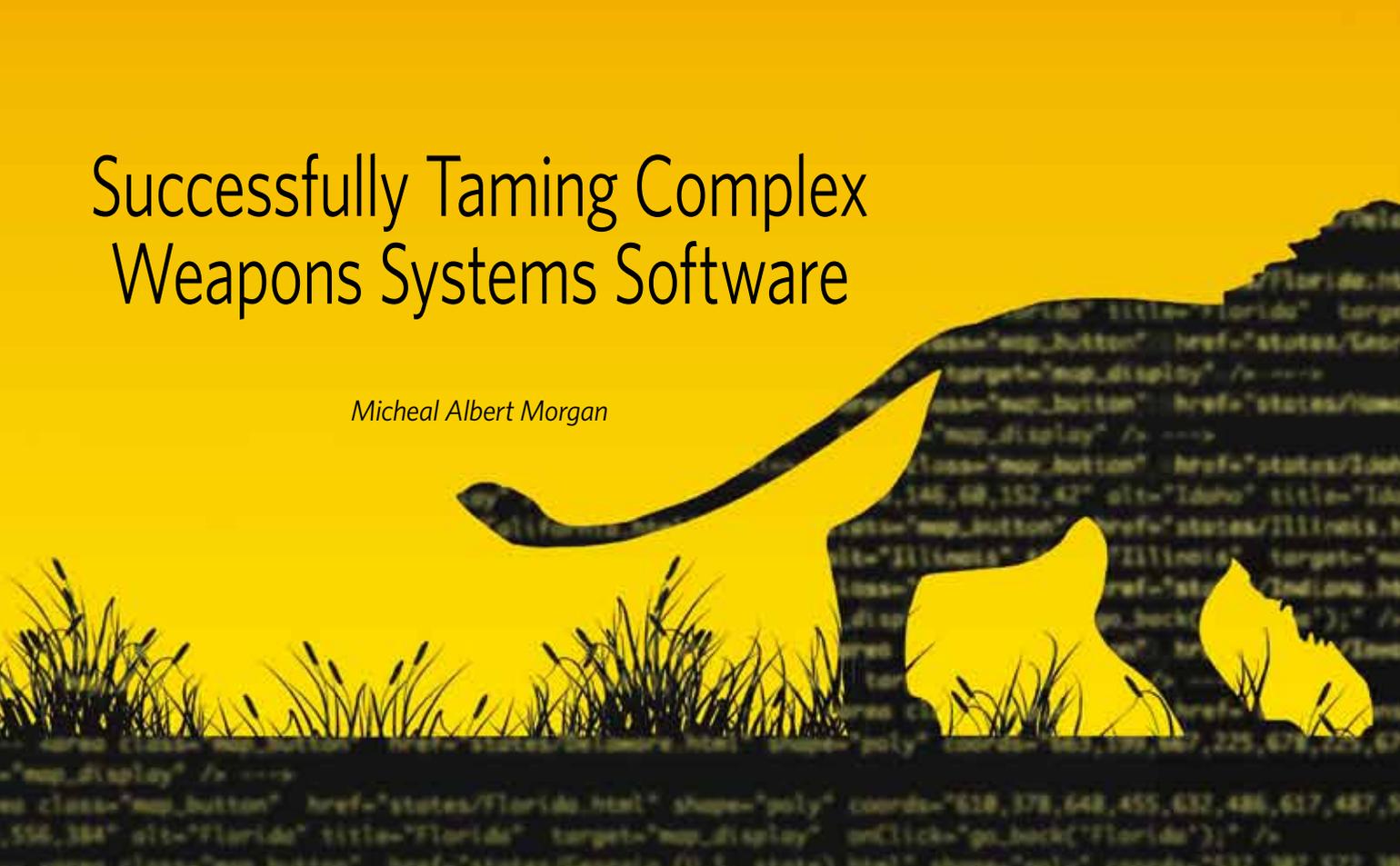# Successfully Taming Complex Weapons Systems Software

Micheal Albert Morgan

Software seems to be one factor that has driven space, aircraft, and other weapons systems to cost overruns and schedule slips—that nebulous "something" we know exists but cannot visualize or get a handle on. Yet it can be tamed, as wild beasts like lions and tigers are tamed by talented animal trainers. I had the privilege of running one medium-sized software system development that was unique in its success. The high fidelity systems simulator (HFSS) is the only project on a contract to endure three Nunn-McCurdy congressional investigations that finished within budget and on schedule.

A complex project requires several thousand small choices. We had the good fortune that the project management office and the contractor facility were within 5 miles of each other. I was always within reach by phone and e-mail and always present for each software coder's computer software configuration item (CSCI) presentations to management, to document CSCI progress. At each review, the coders could ask me questions on topics that would influence the direction of future coding. In turn, I would ask the contractor coders, the other knowledgeable people in the program office, and software testers what features they considered most important.

Communication is crucial to so many human activities, from parenting to leadership in battle. Listening to opinions different from your own and engaging in discussion helps keep the peace. Bridging gaps in perspective is a special skill and a key trait of a leader. However, a decision ultimately has to be made. If a new insight does not sway the disagreeing party, I simply remind the person that it is better to make a wrong decision early in the program, because its wrongness will soon be discovered, and much rework will be prevented.

Morgan *is the systems engineer for the Air Force GPS systems simulator program, previously serving as its project engineer and software maintenance programmer. He has also served in similar positions on other weapon system programs. He has an MS degree in computer science.*

The HFSS team was using the Software Engineering Institute (SEI) personal coding practice, coupled with a team set of practices that had reached Level 4 of capability maturity. Two of the coders had experience as satellite operators, which added an operations viewpoint to the team's system knowledge. The open dialogue among coders, managers, and me (acting as project engineer and project manager) was based on mutual understanding of the critical need for a high-fidelity simulator, for both training and testing. Although we were working amid the onerous atmosphere of the 50 percent cost overruns and 18-month schedule delays that all the other software teams were experiencing, there was camaraderie among the HFSS team members. In addition, the team was determined to show that meticulously following SEI practices could produce a solid product. It was extremely fortunate that the Lockheed Martin management team, the government project manager, and most of the coders recognized the value of SEI and Capability Maturity Model Integration practices.

The team members made a conscious decision to let process rule over preference. The HFSS was composed of more than 1 million source lines of code. One factor that helped HFSS keep on schedule and within budget was writing the test procedures during the coding process (design to test). Additionally, we prevented rework by ensuring we addressed all requirements in the specification. We included satellite software code within the HFSS so the simulated satellite would respond like the actual satellite. Hosting the satellite software turned out to be the most labor-intensive activity of the project; it consumed 40 percent of two of the coders' time. Thus using commercial off-the-shelf or other nondeveloped item code actually takes more schedule than coding the functionality does. We used statistical approaches such as coding response delays, as simple averages of measurements taken over a 2-week period. Using stand-in statistical values when a required or specified value was not supplied allowed coding to proceed while we continued research into actual values. Another example is that the ground-to-space signal delays were made variable based on time of year at each location. For more precisely defined values, such as the simulated hardware, pieces of simulated equipment were coded to behave as described in the technical orders or commercial manuals.

Modeling functionality of an already fielded system may seem far simpler than coding new functionality. It seems natural to assume the HFSS created less risk than the risks that come with the unknowns of a new
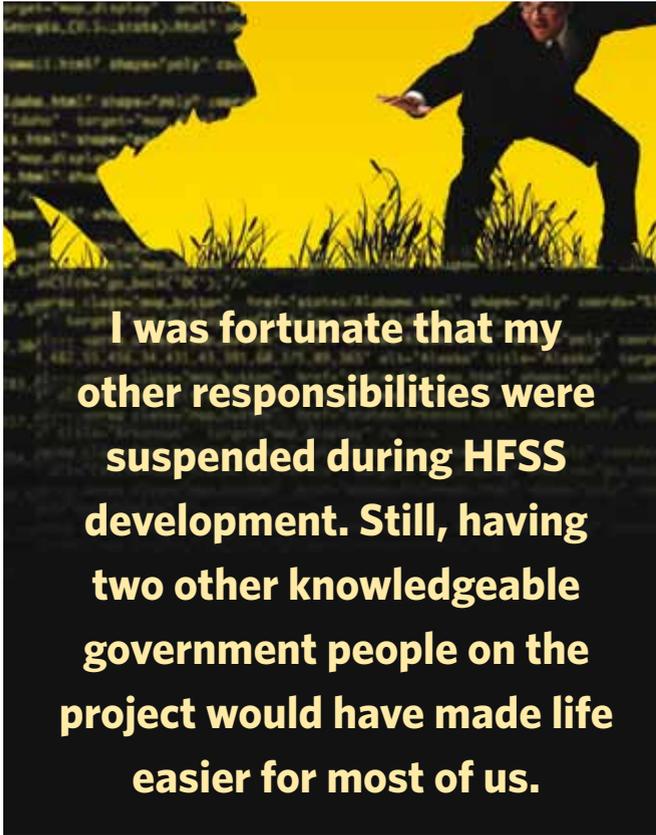
I was fortunate that my other responsibilities were suspended during HFSS development. Still, having two other knowledgeable government people on the project would have made life easier for most of us.

capability. The HFSS was a pioneering project in large-scale space-system simulation. From a technical difficulty viewpoint, I would estimate that the number of open questions was in the 70 percent to 90 percent range of what a similar-sized new capability system would produce. It is apparent to me, from the simultaneous roles I played, that larger software development projects require a government team to properly manage the project, rather than one person. That team should encompass former coders and former operators and be led by a manager dedicated to conformance with best practices. I was fortunate that my other responsibilities were suspended during HFSS development. Still, having two other knowledgeable government people on the project would have made life easier for most of us. This experience leads me to estimate that the acquiring agency needs at least one team member for every 100,000 source lines of code, depending on how the software is arranged.

We also were fortunate to have at our disposal the vast amount of experience and lessons presented in professional society publications. Our experience validated the concept that faithful conformance to best practices, as documented in professional society publications, removes many risk factors. The crippling of innovation and creativity that coders often raise is answered best by reminding them that they can use whatever features the coding language allows to code the process as long as they document how the Information Assurance Workshops standards are met. Of course, a stick and carrot for doing the homework needs to be tailored to each coder. Having the coders commenting and putting hints and other

information in the software-development folder helps when the system is down and the commander is inquiring how long it will take to restore operations. But even for a fix to a routine software problem, it is helpful to have well-documented code.

HFSS won the 2001 Software Simulators Society first prize. Its true value is best demonstrated by the fact it has been updated and in use for more than a decade. Since Boeing replaced Lockheed Martin as prime contractor, the HFSS has been renamed the GPS system simulator (GSS). It has reduced software delivery bugs to less than one per release. It also has reduced operator-induced incidents to zero. This is noteworthy for a system that has become a worldwide utility that industries, military operations, and individuals rely on daily. It has allowed the GPS program office to move beyond the telemetry, tracking, and commanding (TT&C) mission with much more emphasis on warfighter support. No matter how sophisticated the GPS signals become, assurance of no hazardous misleading information is a trust we must strive to fulfill.

A key factor was inclusion of the project manager as a team member in the development effort. I credit the Lockheed Martin project manager for welcoming me into the HFSS team and granting me full access to call on coders and every other member of the project team just as if I were a Lockheed Martin employee. Having been a software maintenance coder, I could understand when coders explained their difficulties. As a systems engineer, I understood satellite functionality and TT&C functionality and how they related. Fortunately, the space-to-control interface was well documented in an interface control document. TT&C functionality was described in a set of hardware and software specifications and design development documents. Satellite functionality was adequately described in a set of orbital operations handbooks, as well as specifications.

HFSS is to date the only project delivered on a contract that actually was reduced in scope (the contract work transferred to the succeeding contract a piece at a time). Upon delivering the HFSS, the Lockheed Martin chief software manager noted: "Nothing helps a project to succeed as well as a well-informed customer." Indeed, knowing software and spacecraft terminology helped me contribute to meaningful conversations on project questions. In addition, the GPS system was quite well documented (I myself had reviewed and edited many of the specifications). That said, the crucial factor in our success was having an experienced and dedicated team of people committed to following SEI well-established performance practices. I must also credit Capt. Lee Corey for taking care of the details of both funding and contracting. He must have worked diligently behind the scenes to shield us from the often critical naysayers who can drag any project into delay. Unfortunately, smooth-flowing, successful projects seem to get little attention and are assumed to be normal by the uninformed. As a taxpayer, I could certainly stand much more "normal" in government acquisition of software. &

The author can be reached at **micheal.morgan@us.af.mil**.