



## Revitalization of Systems Engineering Within the Department of Defense and the Expanding Role of Software



As the Department of Defense (DoD) moves into more complex systems of systems and family of systems with the accompanying challenges of architectures and product lines, software will further dominate development and deployment. Future systems engineers will be required to have more knowledge of software in addition to hardware. This will be a challenging transition for many, yet it is a transition that must be made if we are to be successful in the revitalization of systems engineering for the DoD. The importance of the role of software development in systems engineering processes cannot be overstated.

One of the great misunderstandings of systems engineering and associated processes is the role of software. I believe we are in the midst of a paradigm shift within the DoD. Although developing and building hardware is difficult, we know how to do it. Conversely, few senior leaders or program managers appreciate what is involved in the development and deployment of software. Additionally, not many acquisition professionals recognize the impact of system or requirements changes on software development. These gaps in understanding usually result in software developers being held accountable for schedule slips and budget overruns, which increase alienation of the software community from hardware developers and the systems engineering process.

Across the DoD, significant effort is being invested in the revitalization of the technical work force, especially systems engineering. The focus is on formal education, job training of existing technical staffs, and recruitment of both new and experienced engineers to fill vacancies. While the emphasis is on placing skilled personnel in support of acquisition and in-service engineering programs, there is also the realization that to be successful in recruiting new university graduates, the DoD must offer interesting and exciting hands-on work. Developing and maintaining in-house technical tasks is a priority.

We also need to educate program managers and senior leadership on the software development process, the unique skills of our software work force, and the role of software development in systems engineering processes. In this issue of CROSSTALK, readers will find information that addresses the implications of software's influence on systems engineering now and into the future.

The basics are covered in *Software System Engineering: A Tutorial* by Dr. Richard Hall Thayer where he lays out how software ties the system together. *A Framework for Evolving System of Systems Engineering* by Dr. Ricardo Valerdi, Dr. Adam M. Ross, and Dr. Donna H. Rhodes provides a framework for examining differences between systems engineering and system of systems engineering. Other articles take the reader forward into the software systems engineering world. The article *Using the Incremental Commitment Model to Integrate System Acquisition, Systems Engineering, and Software Engineering* by Dr. Barry Boehm and Jo Ann Lane leads the reader down the path to improved integration of hardware, software, and human factors. Finally, *Systems Engineering for the Global Grid: An Approach at the Enterprise Level* by Patrick M. Kern, *ConOps: The Cryptex to Operational System Mission Success* by Alan C. Jost, and *Issues Using DoDAF to Engineer Fault-Tolerant Systems of Systems* by Dr. Ronald J. Leach take the reader into a much more complex world as we head into the (as yet) uncharted and increasingly complex software systems of the future.

We hold the future success of software systems engineering in our hands. Enjoy these insightful CROSSTALK articles that delve into the world of software systems engineering. I trust you will take insights offered in this issue of CROSSTALK and infuse them into your efforts to develop software and systems engineering of the future.

Dr. John W. Fischer  
Naval Air Systems Command