# Up Close with General Lester L. Lyles

Kathy Gurchiek
CrossTalk *Managing Editor*

CrossTalk *recently caught up with Gen. Lester L. Lyles, Vice Chief of Staff, Headquarters, for the Air Force in Washington D.C. The general talked about the role software has played in the Air Force, and his plans to leverage off of software technology advancements in the new millennium.*

*Lyles entered the military in 1968 as a distinguished graduate of the Air Force Reserve Officer Training Corps program. He received his master of science degree in mechanical and nuclear engineering in 1969 from the Air Force Institute of Technology Program, New Mexico State University. He also is a graduate of the Defense Systems Management College, the Armed Forces Staff College, the National War College, and the National and International Security Management Course at Harvard.*

*His various assignments include program element monitor of the Short-Range Attack Missile in 1974, and special assistant and aide-de-camp to the commander of Air Force Systems Command in 1978. In 1981 he was assigned to Wright-Patterson AFB in Ohio as Avionics Division Chief in the F-16 Systems Program Office. He served as director of tactical aircraft systems and director of the Medium-Launch Vehicles Program and space-launch systems office. In 1992 he became vice commander of Ogden Air Logistics Center, Hill AFB in Utah, where he served until 1994. He commanded the Space and Missile Systems Center at Los Angeles AFB until 1996 and became the director of the Ballistic Missile Defense Organization in 1996. He was promoted to his current position in July.*

*CrossTalk:* What kind of advancements have you seen in the Air Force since you entered it in 1968?

Lyles: Technology has rampantly exploded in almost every mission area for the United States Air Force, and for the Department of Defense in general. Software and software-intensive systems and computer-based systems have been part of the backbone for that growth. Everything we do today is dependent on microprocessors. To some extent software has become the glue that holds everything together. We are totally dependent on good, mature, viable, testable software and I do not think people really appreciate how much software has become the backbone of everything that we are doing.

One of the difficulties we have in software management is we tend to treat it as somewhat ethereal because it is something you do not see. We have some basic management techniques we use for hardware systems, and we forget that sometimes when it comes to software, people do not know how to treat it, they do not understand it. In terms of managing it, we almost have to treat it the same as hardware systems. The fact that it is not visible is sort of a blessing but it also is a curse.

The biggest challenge of some of the contractors that we have dealt with was software development and keeping good software programmers.

*CrossTalk:* One of your goals is to enhance the Air and Space integration in the Air Force. How large a role will software play?

Lyles: Tremendous. When we talk Air and Space or aerospace, and the integration of air and space missions and systems and capabilities, the space realm is the one that probably is more dependent on good software and processing than anything else we do. Since we do not normally have the opportunity—as with space shuttle programs or space shuttle satellites—to operate a man in space, we are dependent on automated systems. Those systems have decision processes or decision processors, and various other computing processors, all of which have the basic software language that makes them operate. As we grow more and more in our space capabilities, dependence on software is going to grow exponentially.

*CrossTalk:* If you had a crystal ball and looked ahead at software and your goals, is there anything specific that you see?

Lyles: Two things. One: software development. Two: Use of processing systems.

All the different mission areas are going to get more complex in the future. We are going to be dependent more on automated systems. Unmanned aerial vehicles, unmanned aerial combat vehicles, seem to be growing significantly in today's age. We are going to be depending even more so on them and without having a man in the loop, rapid processing systems and capabilities are going to be almost mandatory to actually make those things perform properly. Software will be extremely important to make sure we can get them to do what we want them to do on a reliable basis.

I wish that somehow we could break the code in terms of how to manage software development. Before I took over as a

Vice-Chief, I led the Department of Defense's ballistic missile defense system program—often referred to as the old Star Wars program. I cannot tell you how many of our programs were hindered, delayed, over-cost, and way over schedule primarily because of poor software management, poor software development. Contractors always misestimated what it took to develop software.

I have no idea—with all the advances we have made in the last two years—why we cannot figure out exactly how many lines of code you need to do basic functions, develop that capability, get it tested, and then field it as rapidly as we possibly can. I daresay that if you were to do a trace of major problems behind most of our systems that we develop today, software probably will come out—if not the top—very near the top of the factors of why things cannot get done quickly.

The Secretary of the Army [Kenneth C. Royall], went to the entertainment industry to ask it to help develop simulators—training devices—for the things that we are doing or that he is doing in the Army. The things that they are doing for games have a lot better fidelity than some of the things we use to train our troops.

*CrossTalk:* Is it because programs are underestimated in terms of the budget, and in terms of the time needed to complete the project successfully?

Lyles: It is a poor estimation of how many lines of code and how complex the lines of code need to be. And the amount of time to develop those lines of code, get them into a system, get them tested, get the bugs out, and be able to operate it.

Part of that problem for the United States Air Force, and I daresay the entire Department of Defense, is that we are losing software developers rapidly. The commercial industries, entertainment industry, all the other industries out there [need software developers] because more and more depend on process capabilities also.

The biggest challenge of some of the contractors that we have dealt with in the ballistic missile defense program, as an example, was not the hardware development. It was software development and keeping good software programmers. Some companies like Lockheed Martin in Sunnyvale, Calif.—in the heart of the Silicone Valley—had a difficult time keeping good software developers, code developers, and anybody who could test [software]. They were always being lured away by some new start-up company or Internet company.

*CrossTalk:* So how do you keep them?

Lyles: Probably the best solution for us in the Department of Defense is to learn how to take advantage of commercial software a lot more. We all talk about reuse of off-the-shelf software, reuse of software that has already been developed for other purposes. We talk about it, we give it lip service. I am not sure how much we actually take advantage of it. I know we need to do a lot better job in that regard because we just will not be able to attract the kind of people we need to take on that very important function.

*CrossTalk:* What plans do you have to leverage off software technology advancements as we enter the new millennium?

Lyles: We need to do a better job of leveraging commercial software development — particularly code developers [and] software engineers. We are going to have a difficult time motivating people just to come on board to do military-related or defense-related business. We have to do a better job of leveraging the technologies, leveraging the software capabilities for commercial entities, and figure out how we can reuse it for military defense applications. Or modify it slightly so it is not a major redevelopment and recoding [effort] and still get the job accomplished.

*CrossTalk:* With today's trend of outsourcing software acquisition, development, and maintenance functions to contractors, what new roles do you see the 21st century United States Air Force software engineer performing?

Lyles: Management—how to manage in an environment like that. We are already beginning to experience it. I will use the example of our old Star Wars program. We found that we could not attract people to work on defense-related software development programs with a major company, and it had to subcontract its software development to smaller companies, smaller entities in the Silicon Valley area. The company could not afford to hire on and could not retain the software developers in its own company. It had to subcontract that function to small software development houses. That is a whole management entity that is new to all of us—new to industry and new to the Department of Defense. We know how to do it with hardware; it is not unusual to have a major company like Lockheed Martin or Pratt Whitney subcontract to a small vendor to do a part of the development of a hardware piece and then integrate that into a larger whole. We need to figure out how to do that with software also, because we will not be able to depend on a "Lockheed" or a "Boeing" to develop everything internally. They are going to be subcontracting code developments to smaller companies. We need to figure out how to manage that, how to integrate that, to give us the capability we need.

*CrossTalk:* How do you do something like that?

Lyles: I go back to my earlier comment that we tend to treat software as being special—[keeping it] almost at arm's length—in part because people do not understand it. I think we need to take a lesson learned from how we do hardware subcontracting and how we manage that. We need to take the best of those capabilities and apply them to software development.

*CROSSTALK:* How are you planning to use software to retrain the troops?

LYLES: Software is a major part of all of the simulation modeling and simulation systems that we are trying to expand to help us in our education and training activities. As our budgets go down or stabilize, we are finding we are not able to do as many things as we used to, like flying itself. Simulators are becoming more and more important to us in everything we are involved in. I think we are going to see an explosion in the modeling and simulation industry. We have talked about it a lot in the Department of Defense over the last couple of years. We supposedly have some joint modeling and simulation activities under way but they have not really taken hold yet. We will be dependent more and more on modeling and simulation.

The Secretary of the Army [Kenneth C. Royall], went to the entertainment industry to ask it to help develop simulators —training devices—for the things that we are doing or that he is doing in the Army. Going to the [Steven] Spielbergs, going to [George] Lucas, going to those companies to try to take advantage of that. The things that they are doing for games have a lot better fidelity than some of the things we use to train our troops. I love the idea that we are trying to tie-in to the entertainment industry.

The fear is that some of these entertainment companies may not want to help work defense-related programs, so the jury is still out as to whether or not it is going to be successful. But if you are focusing on training and simulation, maybe you would not scare people away who do not want to be involved in war-related activities.

One of the things we are not able to do, or cannot do as much, is fly complex missions [due to] budget, space training areas, and time. If we have a complex mission today that involves several different airplanes of several different types, it is very hard to get all those planes together to train a mission. We are now trying to figure out ways of doing sort of "distributive simulation" and have them all tied-in with very complex software and computer systems so that they, in real-time, can train with each other even though they are all over the country.

We are doing more of that and you will probably hear a lot more of that in the future. It might even become the way we do most of our training. ◆

Software is the language that makes all of that happen. It is really a major part—sort of the unsung hero—of all the things we do and all the missions we try to accomplish.

## What was the best and/or worst software technology innovation of the 20th century?

In preparation for the new millennium, CrossTalk posed this question to its readers. Here are their responses.

### Best

- *the concept of context-free grammars for compilers*
- *WYSIWYG [What You See Is What You Get]*
- *binary computer language and the microchip*
- *databases*
- *MS Word*
- *C++*
- *CD player in personal computers*
- *the computer*
- *Internet*
- *MS Windows*
- *television*
- *e-mail*
- *MULTICS (Multiplexed Information and Computing Service. It is a mainframe timesharing operating system begun in 1965 and still in use today)*
- *computers for scuba diving*
- *machine code*

### W orst

- *Internet*
- *C++*
- *e-mail*
- *World Wide Web*
- *the computer*
- *MULTICS (Multiplexed Information and Computing Service. It is a mainframe timesharing operating system begun in 1965 and still in use today)*
- *MS Word*
- *MS Windows*
- *television*
- *ENABLE software program*
- *databases*