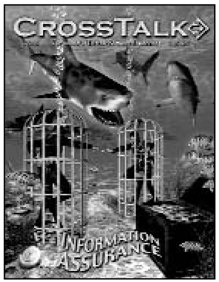
CrossTalk

July 2002 The Journal of Defense Software Engineering Vol. 15 No. 7

Information Assurance



ON THE COVER

Cover Design by Kent Bingham. Shark model courtesy of Alan Roberts.

Software Engineering Technology

Open Forum

Policies, News, and Updates

Departments

CrossTalk

Sponsor	*Lt. Col. Glenn A. Palmer*
Publisher	*Tracy Stauder*
Associate Publisher	*Elizabeth Starrett*
Managing Editor	*Pamela Bowers*
Associate Editor	*Chelene Fortier*
Article Coordinator	*Nicole Kentta*
Creative Services Coordinator	*Janna Kay Jensen*
Phone	(801) 586-0095
Fax	(801) 777-8069
E-mail	crosstalk.staff@hill.af.mil
CrossTalk Online	www.stsc.hill.af.mil/crosstalk
CRSIP Online	www.crsip.hill.af.mil

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address. You may e-mail us or use the form on p. 21.

Ogden ALC/TISE
7278 Fourth St.
Hill AFB, UT 84056-5205

Article Submissions: We welcome articles of interest to the defense software community. Articles must be approved by the CrossTalk editorial board prior to publication. Please follow the Author Guidelines, available at www.stsc.hill.af.mil/crosstalk/xtlkguid.pdf. CrossTalk does not pay for submissions. Articles published in CrossTalk remain the property of the authors and may be submitted to other publications.

Reprints and Permissions: Requests for reprints must be requested from the author or the copyright holder. Please coordinate your request with CrossTalk.

Trademarks and Endorsements: This DoD journal is an authorized publication for members of the Department of Defense. Contents of CrossTalk are not necessarily the official views of, or endorsed by, the government, the Department of Defense, or the Software Technology Support Center. All product names referenced in this issue are trademarks of their companies.

Coming Events: We often list conferences, seminars, symposiums, etc. that are of interest to our readers. There is no fee for this service, but we must receive the information at least 90 days before registration. Send an announcement to the CrossTalk Editorial Department.

STSC Online Services: www.stsc.hill.af.mil
Call (801) 777-7026, e-mail: randy.schreifels@hill.af.mil

Back Issues Available: The STSC sometimes has extra copies of back issues of CrossTalk available free of charge.

The Software Technology Support Center was established at Ogden Air Logistics Center (AFMC) by Headquarters U.S. Air Force to help Air Force software organizations identify, evaluate, and adopt technologies to improve the quality of their software products, efficiency in producing them, and their ability to accurately predict the cost and schedule of their delivery.

Is Our Information Assured?

I don't trust computers. I never have. Online catalogs, automatic teller machines (ATMs), e-bills … computers have become a necessary part of our everyday lives and the need for information assurance is all around us, but the confidence that information assurance exists is not. I got an ATM card soon after they became available. However, I have only used it to withdraw money – I don't trust it enough to hand over my deposit. I also refuse to buy anything over the Internet. I'll browse the Internet but will only place an order if there is a phone number so I can call and talk with a person. I don't like the idea of putting my credit card number into an area that can be hacked.

I understand that all the new automated customer sites are intended to give us faster service at lower prices, but I still prefer dealing with people. However, I find myself being drawn more and more into the computer age and the benefits it provides. Within the Department of Defense (DoD), computing is fundamental to the defense of our country. Dr. Margaret E. Myers, principal director, deputy assistant secretary of defense, made one of several good points during May's Software Technology Conference in Salt Lake City when she quoted Vice Admiral Art Cebrowski: "If you are not interoperable, you are not on the net, not contributing, not benefiting, and you are not part of the information age."

This month's CrossTalk focuses on sharing some of the progress being made in information assurance. We start with an overview of information assurance with Dr. Walter L. McKnight's, *What Is Information Assurance?* This is a good introduction for our readers trying to get familiar with the different security issues.

We get a little more advanced with Julia H. Allen and Dr. Carol A. Sledge's article, *Information Survivability: Required Shifts in Perspective*. These authors share a paradigm shift from merely considering security to a more encompassing focus on survivability. Next is the CERT's Survivable Systems Engineering team's article, *Foundations for Survivable Systems Engineering*, which discusses a methodology developed by the Software Engineering Institute to assess the survivability of a system and make suggestions for improvement. Jim Clune and Dr. Adam Kolawa give more specific advice in *Security Issues with SOAP*. In this article, the authors discuss potential security issues associated with many protocols – while focusing on SOAP as an example – and provide suggestions to overcome these issues.

Peter Baxter then continues our lineup with *Focusing Measurement on Managers' Informational Needs*. Baxter is a recognized leader in the measurement community, and this article provides some back-to-basic ideas when starting or improving a measurement program. Dr. Mario J. Spina and John A. Rolando also share practical advice in *JAD on a Shoestring Budget*. In this article, the authors share their experience when implementing a large Joint Application Development effort.

This issue of CrossTalk would not be complete without reminding our readers of the National Information Assurance Acquisition Policy. The policy goes into full effect this month, requiring acquisition and implementation of only those information assurance products that have been evaluated and validated in accordance with this policy. Readers can learn more about these products by accessing the Validated Products Section of the National Information Assurance Partnership Web site at <niap.nist.gov>.

I still don't like handing out personal information over the Web, but it is reassuring to see the progress being made to compete with hackers. This progress is evident most recently in Afghanistan, where our success thus far is largely because of our informational capabilities to know the enemy's location, our location, and how to proceed accordingly. Obviously, our adversaries have not been able to access this same information. While researching our success with information assurance, I learned that even information about our successes is mostly confidential. However, I have learned enough to realize that even though the proliferation of DoD information capabilities is on a steep upward slope, the compromises to this information continue to be reduced. Our military has the information it needs in the right place at the right time while continually securing and managing that information.

Elizabeth Starrett

Elizabeth Starrett
Associate Publisher

What Is Information Assurance?

Dr. Walter L. McKnight
Shim Enterprise, Inc.

This article defines information assurance from a technical viewpoint, addressing the five attributes of information assurance: availability, integrity, authentication, confidentiality, and non-repudiation. An understanding of information assurance is critical because its activities involve many disciplines, and these activities permeate all phases of software life-cycle development and system maintenance.

Each of us defines information assurance based on our own perspective. For example, a security guard might define information assurance as security clearance and access to buildings or rooms. A network administrator might base his/her definition on passwords or permission rights. Personnel at a network operations center might define it as firewalls, intrusions, viruses, and hackers.

In each of these cases, the parameters of the definition determine who does or does not have access to something. However, information assurance is much more than just that. While each perspective of information assurance is correct, the view of the total picture is not.

From a broad perspective, information assurance includes the products, procedures, and policies that allow the timely transfer of information in an accurate and secure way among all parties involved. While the technology, procedures, and policies used to achieve this have changed over the years, the underlying goals of timeliness, accuracy, and non-repudiation have remained consistent.

For example, in early history man wrote or drew in stone knowing that his neighbors could not easily change it. Timeliness was not an issue because people did not carry stone tablets around with them. By the Roman Empire, man had moved to scrolls that were easier to write on and send. However, scrolls were also easier to copy so seals were created to authenticate the sender. The arrival of the pony express raised the delivery issue; the army was asked to help protect the riders to aid in safe mail delivery.

This article defines information assurance and its terms from a technical point of view. Each term is illustrated for better understanding and will show where the various disciplines associated with information assurance fit into the overall picture. There are concluding references that provide a more in-depth understanding.

Defining Information Assurance

The term information assurance has not been defined in many publications. The definition given in "Information Assurance (IA) Awareness Program," (AFI33-204) is similar to that of the Industry Advisory Council, Shared Interest Group on Information Assurance. They define it as follows: "Conducting those operations that protect and defend information and information systems by ensuring availability, integrity, authentication, confidentiality, and non-repudiation. This includes providing for restoration of information systems by incorporating protection, detection, and reaction capabilities."

> *"Most of the nontechnical staff equates information assurance to information security. This is an incorrect view of the whole picture."*

This same organization defines information security as "the result of any system of policies and procedures for identifying, controlling, and protecting unauthorized (accidental or intentional) disclosure, modification, or destruction of information or denial of service." As you will see, most of the nontechnical staff equates information assurance to information security. This is an incorrect view of the whole picture.

In this article, the terms *information* and *data* are used interchangeably; data assurance is discussed in the same manner as information assurance. While there is a real distinction between the two, it is not the focus of this article.

The term assurance has many meanings. In the context of information, it is defined as a measure of confidence that the security features and architecture of an information system accurately mediates and enforces the defined security policy. This assumes that a security policy has been defined, security architecture has been approved, and security features have been implemented. This confidence is based on analysis involving theory, testing, software engineering, and validation and verification.

For the Department of Defense (DoD), confidence is documented in a System Security Accreditation Agreement (SSAA) that is signed and approved by the designated accreditation authority (DAA) before a system becomes operational. Each system needs to have a signed SSAA.

Lest you think that information assurance is achieved with DAA signing the SSAA, let us now define the five attributes of information assurance: availability, integrity, authentication, confidentiality, and non-repudiation.

Access Means Availability

According to the National Computer Security Center, availability is the "state where information is in the place needed by the user, at the time the user needs it, and in the form needed by the user" [1]. The issues that most directly affect availability are information system reliability (is it up and running?), the informational level of importance (some information is more critical than others), and timely information delivery (delay of some information has a greater impact than other information).

In the past when we wanted information, we had to go to where the information was. We knew where it was located, and we had almost total control of when we wanted to get it. With the development of the Internet, the picture is reversed. We now want information to come to us, and in some cases we want the information as soon as it is generated. Previously, if the information was in another form that we were not familiar with (like a foreign language), it was our responsibility to translate it into a more familiar form. The computer age has changed that, too. Today, we expect tools to be readily available to automatically translate for us. This does not mean going from one foreign language to another but going from a spreadsheet to a word document or a database. We also expect the tools to be accurate 100 percent of the time.

These changes have brought some real challenges regarding information assurance. On the reliability front, we expect our networks, computers, pagers, Palm Pilots, and other information processing devices to work 100 percent of the time. We have built greater reliability into the devices but certain things are not within our control. For example, if someone cuts a fiber optic cable, the network goes down, and we cannot get needed information. To ensure this reliability is maintained, we rely on product designers, maintenance personnel, network designers, network administrators, and helpdesk personnel.

Other information assurance issues affect timely delivery. Sometimes too much information is traveling across the network. On-time delivery becomes a big problem. There are ways to correct this problem, but they involve many disciplines working on all the issues, including software engineers, network engineers, network operations center personnel, and communications engineers. Program managers become involved as well when they address the issue of service vs. cost.

Sometimes the problem is not the amount of required information that transverses the network, but the deliberate introduction of unwanted information into the network. This information creates a problem called *denial of service*. Some people call it *spamming* the network. Some of the disciplines that work on this problem are security personnel, network operations center personnel, security managers, and network administrators.

Some other issues addressed are viruses, worms, and Trojan horses that crash our computers, networks, and other communication devices. This has been a large growth area in information assurance, and many resources are applied daily to make sure these problems do not affect both the reliability of our communication devices and the timely delivery of information.

The last area for availability has been the development of tools that make the presentation of information in the form we want it to be. Typically, we do not think of computer programmers being involved in information assurance, but they play a key role here. Some of the other disciplines involved are requirements engineers, quality assurance personnel, and configuration managers.

Integrity

The second IA attribute is integrity, which is "sound, unimpaired, or perfect condition" [2]. Here we are looking more at system integrity instead of data integrity (although both can be considered).

System integrity looks at the overall architecture of the system and how it is implemented. The design has to follow best practices and considers how various devices affect the overall design. You would not want to put a chokepoint into a network that would become a prime target for an enemy to hit or which could bring down the network if it became inoperable.

> *"Authentication ensures that you have the right to see the information, and that you are who you say you are … not only do people need to be authenticated, so do devices."*

Not only are we concerned with how the system is designed, but also how it will be maintained. We do not want to make it more costly to maintain than what the information is worth. We also look at what happens when unforeseen events happen, such as earthquakes or power failure. We need to have contingency plans in place so we can continue our operations during these types of events. Some of the disciplines involved here are program managers, system designers, contingency planners, operations personnel, and human resources personnel.

Authentication

The third IA attribute is authentication, which is defined by the National Computer Security Center as follows: "… 1) to verify the identity of the user, device, or other entity in a computer system, often as a prerequisite to allowing access to resources in a system, and 2) to verify the integrity of data that have been stored, transmitted, or otherwise exposed to possible unauthorized modification" [3]. Authentication ensures that you have the right to see the information, and that you are who you say you are.

The two elements often associated with authentication are logins and passwords. You are generally given a login name when the system administrator is sure you are who you say you are. You then establish a password so the system can be sure you are who you say you are. Helping the system administrator is the security personnel who might look into your background to see who you really are. Instead of a login name and password, a fingerprint device or retinal scanner may help establish who you are.

Not only do people need to be authenticated, so do devices. The network might need to confirm where it is getting its information, and it may require that routers, bridges, and other communicating devices identify themselves to the network. These network devices go through a process of exchanging information to establish their identity. This *authentication process* is often called a network protocol. Here, a different group of individuals are involved such as network programmers and standards committees who determine what valid protocols are, and how they are to be implemented.

Authentication also makes sure that the needed information is not altered between the time it is generated and the time it is received. There are several ways information can be altered, including viruses, worms, or Trojan horses that alter information at any time during generation, transmission, or receipt. For example, an unscrupulous individual monitoring the network could change the information while it is traveling across the network. Or, some of our translation tools could introduce errors into the information.

Some of the disciplines involved in making sure information integrity are maintained include network operations personnel who are looking for unusual activities on the network. Security police might patrol unsecured portions of the network. System designers might include a protected distribution system to make sure intruders cannot get into the network. Software testing personnel could conduct tests to make sure the translation tools work as designed. Configuration management personnel could ensure that the right version of the software is operational on the system.

Confidentiality

Confidentiality is the fourth IA attribute. It is "the concept of holding sensitive data in confidence, limited to an appropriate set of individuals or organizations" [4]. Confidentiality is often referred to as information security. Here we deal with two issues: clearances and data security.

Access to data is based on two criteria: a security clearance and a *need to know*. In the DoD, there are several agencies whose mission is to determine the trustworthiness of an individual. Security clearances are issued based on that trustworthiness. Normally, security personnel only deal with those individuals who need and/or have a security clearance. It is up to the data owner to determine who has the need to know. The disciplines involved here utilize security managers, investigation personnel, arbitrators, operational personnel, and human resources personnel.

Data security can be provided by building private networks, encrypting the data that travel across unprotected sections of the network, providing protective distribution systems, or building secure enclosures where the data can be processed. These measures use security personnel, communications security personnel, emanation personnel, program managers, communications engineers, and a dozen other disciplines. The National Security Agency becomes involved in the many issues associated with data security.

Non-Repudiation

The last IA attribute is non-repudiation. This is "a service that provides proof of the integrity and origin of data, both in an unforgeable relationship, which can be verified by any third party at any time; or, an authentication that with high assurance can be asserted to be genuine, and that cannot subsequently be refuted" [5].

There are three types of services in non-repudiation: non-repudiation of origin, non-repudiation of submission, and non-repudiation of delivery. Non-repudiation of origin protects against any attempt by the message originator to deny sending a message. Non-repudiation of submission protects against any attempt by message transit point to deny that a message was submitted for delivery. Non-repudiation of delivery protects against any attempt by a message recipient to deny receiving a message. Two of the services that support non-repudiation are data signature and encryption.

Data signature is a fairly new area of information assurance, although ideas for it have been around for a long time. The technology for doing data signatures is still not at the level of confidence that is needed for widespread use. Many legal issues need to be addressed, which utilize two other professions to IA: lawyers and judges.

We have already addressed the issue of encryption. Some new technologies are emerging that will make data encryption less costly and less man-power intensive. Much of this new technology has been developed because e-commerce has created a greater need for encryption.

> *"Confidentiality is often referred to as information security. Here we deal with two issues: clearance and data security … Access to data is based on two criteria: a security clearance and a need to know."*

Several disciplines are involved in IA. Today, IA is an issue that has to be addressed in every phase of a system life cycle. Even in system disposal, information assurance plays a key role. After years of protecting information, you do not want to give it all away with an improper system disposal.

The key document ensuring that all attributes of IA are addressed is the SSAA. This document begins with the concept design phase and is reviewed regularly to make sure that any changes to the system have addressed security issues.

Summary

Information assurance and its attributes have been defined in both technical and nontechnical terms. The author has only brushed lightly across some of the issues associated with IA. A great general source on information assurance for program managers and others who want a general concept review is, "An Introduction to Computer Security: The NIST Handbook" [6]. More technical documents have been developed by the National Computer Security Center (both technical reports and technical guides) and by the National Institute of Standards and Technology (both their own special publications and the Federal Information Processing Standards publications).

We are all involved in information assurance. Not only do we depend on it to do our work, but also we are involved in making sure it works. Remember, information is only as good as the assurance that we apply to it. Not all information needs to be protected at the same level, but all information needs to be protected.◆

References

1. National Computer Security Center. "Glossary of Computer Security Terms." NCSC-TG-004-88. Oct. 1988: 9.
2. Ibid, p. 23.
3. Ibid, p. 8.
4. Ibid, p. 12.
5. Caeli, W., D. Longley, and N. Shain. Information Security Handbook. London: Macmillan, 1991.
6. NIST Special Publication 800-12. An Introduction to Computer Security: The NIST Handbook. Oct. 1995.

About the Author

Walter L. McKnight, Ph.D., is a senior information assurance engineer at Shim Enterprise, Inc. where he is conducting security accreditation for the Ground Theater Air Control Systems. He has more than 30 years of experience as an Air Force officer working in all phases of computer technology. Dr. McKnight has a bachelor's of science degree in mathematics, a master's degree in computer science from the University of Utah, and a doctorate in computer science from Ohio State University.

1732 Westerly Drive
Brandon, FL 33511
Phone: (813) 643-7343
E-mail: wmcknigh@shiminc.com

Information Survivability: Required Shifts in Perspective

Julia H. Allen and Dr. Carol A. Sledge
Software Engineering Institute

Organizations today are part of an interconnected, globally networked environment – one that continuously evolves in ways that cannot be predicted. What effect does this environment have on the survivability of the mission of an organization? To improve survivability, organizations must shift their focus from a more information security-centric perspective to one that includes an information survivability-centric perspective.

The events of recent years and especially of recent months have greatly increased awareness of information and infrastructure security, whether they are media reports of the latest cyber attacks and vulnerabilities or postulations as to the degree of permeability of our critical infrastructures.

While this may spark reactions such as reviews of organizational computer security policies and vulnerability assessments, attention to issues of security, while important [1], cannot ensure the preservation of mission-critical services when systems are penetrated or compromised. Survivability, an emerging discipline, incorporates a new technical and business perspective on security, creating solutions that focus on elements such as the continuity of critical services.

In terms of solution space, security takes a technology centric point of view, with each technology solving a specific set of issues and concerns that are generally separate and distinct from one another. Survivability takes a broader, more enterprise-wide point of view looking at solutions that are more pervasive than point-solution oriented.

Survivability

We define survivability as "the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents" [2]. A survivability approach combines risk management and contingency planning with computer security to protect highly distributed information services and assets in order to sustain mission-critical functions. Survivability expands the view of security from a narrow, technical specialty understood only by security experts to a risk management perspective with participation by the entire organization and stakeholders.

To improve the survivability of the organization's mission, senior management must shift its focus and that of the organization from an information technology (IT)-based, security-centric, technology solution perspective to an enterprise-based, survivability-centric, risk management perspective. Experience in our executive workshop[1] has shown that many do not know how to think about information survivability in a useful way, or understand the role they should play in promoting survivability.

Seven Shifts in Perspective

We have observed seven shifts in perspective or shifts in thinking that we believe are essential to move from an IT-based, security-centric, technology solution point of view to one that is more enterprise wide, based on survivability and builds on risk management (Table 1).

For each of these seven shifts in perspective, we describe some example indicators. The presence or absence of these indicators may give some notion of whether or not the shift is in progress, or if it has actually occurred. We do not claim these indicators are definitive or comprehensive, but merely exemplars. Similarly, we present examples of questions senior management can ask to elicit the current state of the organization.

Asking the right questions is essential for senior management to understand the critical role that survivability plays in fulfilling its mission and objectives, as well as the risks that need to be managed [3]. Creating organizational awareness about survivability is essential for it to be factored into key decisions. This assumes that mission and information survivability are high priorities when weighed against other pressing priorities that vie for senior management's attention.

Shift 1: Central to Global

The first shift in perspective is from systems that are in a centrally networked environment under organizational control with full visibility, to systems that are in a globally networked environment with no bounds, no central control, and limited visibility into the systems. Physically isolated, stand-alone mainframe or corporate environments have evolved into a distributed client server network that are connected to the Internet with peer-to-peer services and networking. It is no longer the case that access is permitted only within the physical facilities that house the network: Remote access is now a given.

This shift in perspective may be indicated by actions taken to regularly evaluate and address key risks to key assets based on global access and often unknown threats from unidentified sources. It may also be indicated by the presence of a network/system architecture where critical assets (including functions/services) are distributed and stored redundantly [4].

Questions to initiate or indicate this shift include the following:

- Is the frequency and scope of the organizational risk evaluation sufficient to evaluate key risks to key assets

Table 1: *Seven Shifts in Perspective*

FROM	TO
Systems are centrally networked, under organizational control.	Systems are globally networked with distributed control.
Systems are bounded with defined geopolitical boundaries.	Systems are unbounded with no geopolitical boundaries.
Clear distinction between insiders and outsiders.	Often cannot distinguish between insiders and outsiders.
Predictable processing load and events.	Unpredictable load and asynchronous events.
Organizational responsibility.	Distributed responsibility.
Security as an overhead expense.	Survivability as an investment; essential to the organization.
Technology, IT-based solutions.	Enterprise-wide, risk management solutions.

and take into account evolving threats?

- Does the continuity plan sufficiently address how to protect the confidentiality, integrity, and availability of critical assets?
- Is the security policy sufficient and effectively enforced for today's globally distributed environment?

Shift 2: Bounded to Unbounded

The second shift in perspective is from systems that have well-defined geographic, political, cultural, and legal or jurisdictional boundaries, to systems characterized by the absence of these boundaries. Centralized administrative control with trustworthy, known, inside users evolves to systems with distributed administrative control without central authority and unknown users. This shift is also indicated by the presence of an active network of administrators with time to stay up-to-date, stay connected, and stay in communication with one another.

Questions to initiate or indicate this shift include the following:

- Do strategic and tactical security decisions derive from an appreciation that networks, when connected to the Internet, have no well-defined geographical, political, and technological boundaries?
- Do system and network administrators have an active contact list of peers for the primary networks interfaced?
- Are administrators up-to-date on the latest threats, attacks, and solutions?
- Are system and network configurations up-to-date with the latest patches?

Shift 3: Insular to Networked

The third shift in perspective is from viewing systems as insular and fortress-like, to viewing systems as being networked and interdependent; the ability to distinguish between insiders and outsiders decreases. Outsider roles go from being well-defined to the realization that an outsider can be a customer, collaborator, partner, contractor, or vendor; outsider access to the network changes based on that role. Do we have layered security architecture (*defense in depth*), understanding that organizational perspective shifts from thinking a firewall will protect the network to the realization that a firewall is just one part of layered security architecture? In-house infrastructure maintenance may shift to the outsourcing of all or part of the infrastructure and may include managed security services (e.g. firewalls, intrusion detection monitoring, incident response, and penetration testing).

This shift may be indicated by the presence of a decision process allowing third-party access, with active management of each type of relationship with the appropriate level of security. Secure means exist for remote access, authentication, and access control; virtual private network technologies may be used. Accounts are retired when partnerships or relationships terminate.

Questions to initiate or indicate this shift include:

- Do we have a layered security architecture?
- Are there decision processes and supporting procedures to permit third-party access and to manage each type of relationship with the appropriate level of security?
- Do we understand and implement appropriate security controls for managed security services provided by outside parties?

> *"Outsider roles go from being well-defined to the realization that an outsider can be a customer, collaborator, partner, contractor, or vendor; outsider access to the network changes based on that role."*

Shift 4: Predictable to Asynchronous

The fourth shift in perspective is from one where processing events happen in predictable, prescribed sequences and patterns with predictable loads, to one where events often occur asynchronously, independent of time sequence with unpredictable loads. The situation becomes one where anything can happen anytime: Work proceeds 24 hours a day, seven days a week, and distributed denial-of-service agents can be installed and launched at any time.

A clear understanding and management of risk where predictability is important indicate evidence of the shift. It may be necessary to take these particular processes offline, to create an *air gap*. The shift is also manifested by diligence to ensure installed attack agents are detected and eliminated.

Questions to initiate or indicate this shift include:

- Are processes and transactions that need to occur in a predictable sequence sufficiently protected from disruption?
- Do administrators regularly scan for the presence of denial-of-service agents?
- Is the integrity baseline maintained and regularly checked for all critical assets?

Shift 5: Single Responsibility to Shared Responsibility

The fifth shift in perspective progresses from single responsibility to shared organizational responsibility to distributed responsibility. This is a shift from having a single point of known responsibility to correct failures, to having shared sometimes unknown responsibility. In other words, going from, "I know who to contact when I have a problem and I can describe the problem" to a situation better described as, "I cannot precisely identify what or where the problem is, and I may not know who to contact if it occurs outside of my organization's administrative control."

The shift is indicated by everyone knowing who to call first inside of the organization, with the responder performing triage on all calls. That responder relies on his/her contact list for assistance and solutions. Those collectively responsible understand their high degree of interdependence and are quick to assist.

Questions to initiate or indicate this shift include:

- Do all authorized users know whom to contact when they detect suspicious, unexpected, or unusual behavior?
- Do the recipients of this information know how to process each request, dealing with highest priority requests first, and know who to contact for further assistance?

Shift 6: Overhead to Essential

The sixth shift in perspective is from viewing security as an overhead activity and expense, to viewing survivability as an investment that is essential to the organization, along with ensuring that there is always a contingency plan. It reflects a change of view. Instead of security being IT's responsibility, with IT and the CIO constantly having to justify their budget for security, survivability is regularly reviewed and discussed in senior-level management meetings and is accepted by all as part of being in business.

Questions to initiate or indicate this shift include:

- Is the term *survivability* an active part of the vocabulary at all organizational levels?
- Is survivability regularly reviewed and discussed in senior-level management meetings?
- Is work to sustain/improve security and survivability a standing budget line item that does not require annual justification?
- Do continuity and disaster recovery plans adequately address security and survivability concerns? Are these plans regularly tested?

Shift 7: Security to Survivability

The seventh shift in perspective is from technologic IT-based solutions to enterprise-wide, risk-management solutions. Instead of viewing security as a narrow, technical specialty accessible only to experts and focusing on the protection of specific components, survivability is embraced as a risk-management perspective that requires involvement of the whole organization and focuses on the survival of the mission rather than a particular component.

Senior managers must change their view that "protecting the network is a matter of listening to the right experts and installing the right technology solutions." Rather, their declared view is that "the survival of the mission depends on the ability of the network to provide continuity of service, albeit degraded, in the presence of attacks, failures, or accidents."

The shift is indicated by the absence of silver-bullet thinking. It is replaced by understanding that this is a long-term, continuous activity required for the success of the organization. In other words, senior management needs to think of survivability and its contribution to the organization the same way that they would think of any critical organizational process or organizational function that they perform (such as meeting profit objectives, growing through acquisition, and raising stockholder share value). Survivability must have the same importance and receive the same level of attention as any of those other key processes.

Questions to initiate or indicate this shift include the following:

- Are security and survivability risks managed as actively as other risks?
- Is it understood (as manifest in our speaking and actions) that the survivability of the infrastructure is essential to the survivability of the organization and mission?
- Are IT staff members involved in executive and management-level decisions on security and survivability and vice versa?

Summary

Given that more and more of today's organizations are part of an interconnected, globally networked community, this shift in thinking is imperative. The survivability of an organization's mission requires that senior management and their organizations shift their thinking from an IT-based, security-centric, technology solution point of view, to one that is more enterprise-wide, based on survivability and that utilizes risk management approaches. As a start, for each of the seven shifts in perspective, think about where your organization is today: Has it already accomplished the shift? Is it in progress? How might this shift be initiated? ◆

References

1. Allen, Julia, Christopher Alberts, Sandi Behrens, Barbara Laswell, and William Wilson. "Improving the Security of Networked Systems." CrossTalk Oct. 2000.
2. Lipson, Howard, and David Fisher. "Survivability – A New Technical and Business Perspective on Security." Proceedings of the 1999 New Security Paradigms Workshop. Association for Computing Machinery. New York, 1999. Available at <www.cert.org/archive/pdf/busperspec.pdf>.
3. Allen, Julia H. "Ask the Right Questions." Internet Security Alliance. 26 Oct. 2001. Available at <www.isalliance.org/working/practices.phtml>.
4. Linger, Richard C., Robert J. Ellison, Thomas A. Longstaff, and Nancy R. Mead. "The Survivability Imperative: Protecting Critical Systems." CrossTalk Oct. 2000.

Note

1. "Survivability: A New Executive Perspective" is a course offered by the Software Engineering Institute, Carnegie Mellon University, Pittsburgh.

About the Authors

Julia H. Allen is a senior member of the technical staff of the Software Engineering Institute (SEI). Her work includes the development of security improvement practices for network-based systems. She previously served as SEI acting director and deputy director. Before joining the SEI, she was vice president of Science Applications International Corporation (SAIC), where she was responsible for starting a division that specialized in embedded systems software. She recently published *The CERT Guide to System and Network Security Practices.*

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-3890
Phone: (412) 268-7995
Fax: (412) 268-7966
E-mail: jha@sei.cmu.edu

Carol A. Sledge, Ph.D., is a senior member of the technical staff of the Software Engineering Institute (SEI). Her work includes executive education and the investigation of practices and strategies for survivable enterprise management. Dr. Sledge previously investigated commercial off-the-shelf based and open systems. Before joining the SEI, she managed the acquisition, development, and support of large multiplatform, system-software product lines at a number of corporations. Dr. Sledge has developed and taught a variety of software engineering and computer science courses.

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Room SEI/SP 406
Pittsburgh, PA 15213-3890
Phone: (412) 268-7708
Fax: (412) 268-7966
E-mail: cas@sei.cmu.edu

Foundations for Survivable Systems Engineering

Dr. Robert Ellison, Richard Linger, Dr. Howard Lipson, Dr. Nancy Mead, and Andrew Moore
Software Engineering Institute

The complexity of today's large-scale networked systems increases their vulnerability to intrusion, compromise, and failure. We are addressing the survivability of these systems by establishing new methods for risk assessment and by developing engineering technologies for analysis and design of survivable systems.

Survivability of critical infrastructure systems has become an urgent priority. These large-scale networked systems improve the efficiency of organizations through new levels of integration and communication. However, increased integration is accompanied by increased risks of intrusion, compromise, and cascade failure effects. Incorporating survivability into these systems can mitigate these risks.

Survivability focuses on preserving essential services, even when systems are penetrated and compromised [1]. As an emerging discipline, survivability builds on related fields of study (e.g., security, fault tolerance, safety, reliability, reuse, verification, and testing) and introduces new concepts and principles.

Survivability is defined as "the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents" [2]. The term *system* is used in the broadest possible sense to include networks and large-scale systems. A key observation in survivability engineering is that no amount of security can guarantee that systems will not be penetrated and compromised. The complexities of Web-based services, issues of function and quality in commercial off-the-shelf (COTS) usage, and the proliferation of end-user devices and channels, combined with the growing sophistication of attacks and intrusions, present formidable engineering challenges in survivable system analysis and development.

Attacks exploit not only specific system vulnerabilities but also trust relations between systems. Attacks can target networks, devices, and user task flows. Sophisticated intruders include cyber-terrorists, non-state activists, and state-sponsored adversaries (foreign intelligence services and militaries), as well as insiders. Sophisticated intrusions are becoming more likely and more difficult to counter.

Many attacks target vulnerabilities in system components such as domain name servers or Web servers. Boundary control mechanisms, such as firewalls and demilitarized zones, provide some defense against these attacks. But it is often the case that security is addressed too late in the development cycle, with boundary controllers used for after-the-fact remediation when systems are deployed. Moreover, the adequacy of boundary controllers decreases as user task flows traverse multiple system boundaries and security administration domains. Sophisticated intruders can attack a broad range of targets across domains. Resistance and response to such attacks are often the responsibility of multiple enterprises and their system and application architectures.

> "A key observation in survivability engineering is that no amount of security can guarantee that systems will not be penetrated and compromised."

Security typically focuses on what is regarded as well-defined boundaries and control of internal components and systems within those perimeters. The reality of today's large-scale network systems is quite different. User task flows, system boundaries, and user communities are dynamic and difficult to analyze. The question of where or how to define the system boundary becomes highly important when considering survivability. The old notions of system boundaries may not fit the current environment.

Web services, although seemingly innocuous, may provide an opportunity for an attack. Remote access to systems such as that afforded by cable modem connections may also enable attacks. Any facility that provides an opportunity for attack on your system should be considered when performing a survivability analysis. Task flows cross multiple system and organizational boundaries and exhibit dependencies on external systems and on COTS components. New Web service and network communication infrastructures support such flows. And open-distributed architectures present whole new categories of vulnerabilities. These system realities drive two key problems in survivability design and development:

- How to design survivability into highly distributed systems despite limited central administration, poor visibility of end-to-end task flows and system dependencies, and dynamic functionality and usage.
- How to manage survivable system evolution in terms of changes in functional requirements, threats, and operating environments.

The CERT Survivable Systems Research Agenda

We believe that new engineering methods are required to deal with these problems within the realities of today's dynamic, network-centric systems. Our research is aimed at theoretical foundations, language representations, and rigorous yet practical unified engineering methods to represent and reason about systems, their (often COTS) components, and their threat environments. Much of our work is documented in publications that can be found and downloaded from the CERT Web site at <www.cert.org>, particularly in the pages on "Survivable Systems Engineering" at <www.cert.org/sna>. Our overall objective is to improve system engineering practices for survivability. Such practices require solid engineering foundations.

For each life cycle activity, survivability goals should be addressed and methods to ensure survivability incorporated. If addressed at all, survivability issues are often relegated to a separate thread of project activity, with the result that survivability is treated as an add-on property. This isolation of survivability considerations from primary system development tasks results in an unfortunate separation of concerns. Survivability should be integrated and treated on a par with other system properties to develop systems with required functionality and performance that can also withstand failures and compromises. A survivability baseline needs to be established fairly early on, for example, during the development of concept of operations, and revisited at major development milestones such as requirements baseline, architecture baseline, etc. This sounds as if it suggests a waterfall-type life cycle, but in fact works nicely with more modern life cycle models such as the spiral model.

In some cases, existing development methods can enhance survivability. Current research is creating new methods that can be applied; however, more research and experimentation is required before the goal of survivability can become a reality. Our research agenda has its roots in the CERT (formerly known as Computer Emergency Response Team) Survivable Systems Analysis (SSA) method (formerly called Survivable Network Analysis) that we have been applying with clients for several years. Although we do not have documented cost/benefit data, in most cases it is clear to the clients that our recommendations will improve the survivability of their systems, thus the implementation decision is relatively easy to make.

SSA is a structured engineering process aimed at improving survivability characteristics of new or existing systems. A small team of survivability experts working with a client team of subject matter experts conducts it. SSA is carried out in a series of joint working sessions, and the findings are summarized in a report for management action [3]. The SSA process begins with briefings from system users, stakeholders, and developers typically focused at the architecture level. The discovery process continues with developer, user, and stakeholder views of essential services and assets of the system, that is, the services and assets that must be available no matter what the threat environment and state of compromise. These services are formulated as stepwise usage scenarios and traced through the architecture to reveal corresponding essential components.

Next, representative intrusions are identified based on analysis of the threat environment and, likewise, expressed as usage scenarios for tracing through the architecture to reveal components that can be compromised. With this information, it is possible to identify soft spot components that are both essential and able to be compromised, followed by survivability analysis for improvements to resistance, recognition, and recovery strategies within the system architecture. It is often the case that recommendations propagate to areas such as requirements, policy, and operations. Our application of SSA with clients has resulted in three key observations that drive the research agenda:

- Systematic evaluation methods are required for assessing COTS component survivability. Many organizations are developing mission-critical systems using COTS components. COTS can offer lower, up-front costs than custom-built solutions, but acquiring organizations lack access to the artifacts of the software engineering process used to create the components. Analysis of engineering artifacts is the traditional means for verifying the survivability of custom-built systems. One way to partially compensate for this lack of access is to use a vendor-risk assessment as a tool in building, maintaining, and evolving survivable systems. We are developing a risk-management approach called Vendor Risk Assessment and Threat Evaluation (V-RATE) [4] for assessing the survivability of COTS-based systems. V-RATE assessment helps acquiring organizations to understand the trade-offs associated with using COTS products, and to achieve the required assurance levels through evaluation and interaction with COTS vendors. It also supports comparison of different system designs based on alternative COTS products.
- Large-scale network system complexities can be reduced and managed by a unified engineering discipline for analysis and design that includes survivability in a comprehensive framework. Complexities of large-scale network system analysis and design often exceed engineering capabilities for intellectual control. We are defining engineering foundations for Flow-Service-Quality (FSQ) technology [5] based on user task flow structures and their architecture traces, a computational approach to quality attributes (including survivability), and an architecture framework for dynamic management of flows and their quality attributes. This process can be applied to specification, design, and operation of new systems, as well as to analysis of existing systems for survivability dependencies and risks that can impact mission performance. It also assists in integrating stovepipe systems to support new mission objectives.
- Structured documentation and systematic use of attack patterns and survivability strategies can help design and analyze intrusion-resistant architectures. Major investment in information security technology by a business or military enterprise often translates into little, or questionable, value to the operational mission. A primary reason is that many design and analysis efforts focus on deciding which popular security technologies to integrate, rather than on a rational assessment of how to address attacks that are likely to compromise the mission. Our work involves incorporating intrusion and risk-analysis techniques into existing development practices. This work requires consideration of the larger operational context in which system technology resides, which we call the *enterprise*. Enterprise architectures need to be developed and analyzed just like the systems on which they are based. These research projects are discussed in detail below.

> *"A survivability baseline needs to be established fairly early on ... during the development of concept of operations and revisited at major development milestones."*

Vendor Risk Assessment and Threat Evaluation Project

Building survivable systems using COTS components is a daunting task because the developer has little or no access to the artifacts of the software engineering process used to create the components. These artifacts are the primary sources from which assurance evidence for a composite system is derived. One way to partially compensate is to use vendor risk assessments as a tool to help build, maintain, and evolve survivable systems. Such an assessment can be used as a new source of assurance evidence of a system's survivability.

Our vendor risk assessment approach, V-RATE, is based on the taxonomy described in Table 1. Two broad categories are at the highest level of the taxonomy: 1) vendor-inherent risk elements, and 2) vendor-risk elements that are associated with your own risk management skills. The output of an assessment based on the V-RATE taxonomy is a vendor-risk profile for the system being evaluated. We envision a large and growing collection of ven-

Vendor's Inherent Risk Elements	
Visibility of Product Attributes	Openness - degree of visibility into design and engineering processes. Independent testing organizations.
Technical Competence	Survivability capability maturity. Existence of vendor ratings/certifications. Evidence of adherence to applicable industry standards and government regulations. Demonstrated diversity and redundancy in a vendor's products and services. Existence of a vendor team that deals effectively with security/survivability issues.
Performance History	Evidence that demonstrates a track record of dealing successfully or unsuccessfully with survivability issues and events.
Compliance	Responsiveness to security/survivability issues (which can include related quality issues such as reliability, performance, safety, and usability). Responsiveness to requests for new features and improvements. Willingness to cooperate with third-party testers and certifiers.
Trustworthiness	Track record/word-of-mouth. Evidence of skill at evaluating trustworthiness of personnel, e.g., the vendor consistently checks the character references of new hires and periodically re-checks all personnel.
Business Management Competence	Economic viability. Vendor's risk management skills in dealing with subcontractors.
Controlled Evolution	Clearly specified (or discernible) evolutionary path. Product integration stability. Product evolution supports continual survivability improvement.
Vendor Risk Elements Associated With Your Risk Management Skills in Dealing With Vendors	
Technical Risk-Mitigating Factors	Your skill at evaluating a product's quality attributes (in particular, those quality attributes that can contribute to system survivability such as security, reliability, performance, safety, and usability). Your skill at evaluating vendor technical competence. Your awareness of existing vendor ratings and certifications. Demonstrated diversity and redundancy in the integration of vendor products and services. Use of architectural tools and techniques (e.g., wrappers) to limit risks associated with a vendor product. Your association with expert security/survivability organizations and the existence of a dedicated security/survivability group within your own organization.
Nontechnical Mitigation of Risk	Legal (e.g. license agreements). Economic (e.g. insurance). Political and social (e.g. regulatory protection).
Independence/Interdependence	You examine the vendor products and services associated with your system and look for interdependencies that could threaten survivability.
Your Exposure	You determine what elements of your system are dependent upon the competence, trustworthiness, and thoroughness of the vendor.
Mission Alignment/ Vendor Compatibility	You evaluate the alignment of your mission and the required software quality attributes (SQAs) with the vendor's mission and SQAs.
Your Negotiating Skill/ Bargaining Power	Use of economic or other leverage to obtain vendor concessions that enhance survivability such as early notification of security vulnerabilities.

Table 1: *The V-RATE Taxonomy*

dor-risk profiles tied to real-world performance histories, providing empirical data against which a newly generated risk profile can be compared. A vendor-risk profile can be used to assess the risk associated with the use of a product in a particular threat environment and to identify areas for additional risk-mitigation activities. Because a single numerical rating would not provide sufficient guidance for these risk-mitigation activities, the vendor-risk profile helps identify your risks in each of the V-RATE taxonomy areas and allows you to consider your risk tolerance with respect to each element of the taxonomy.

We need to apply the V-RATE method to real-world, mission-critical systems. Such case studies will help us fine tune and validate the method and demonstrate its use within a realistic life cycle process. These studies will also help us to understand the risks associated with using COTS components for specific system missions. Details of the application of V-RATE (such as the specific evidence that needs to be gathered) may differ for different domains (e.g., military mission-critical, e-commerce, and financial systems). Since survivability is heavily dependent upon the context of the mission, understanding these differences is critical to V-RATE's successful application. We have an immediate plan for conducting a case study of the V-RATE method with a Carnegie Mellon University project in the coming months.

Flow-Service-Quality Engineering Project

Imagine the flow of communications and operations among networked systems that support the simple task of purchasing gasoline with a credit card. The purchaser must enter input data. Communications must be established with the credit card organization, perhaps through a combination of land lines and satellite links. Credit databases and business rule services must be accessed, perhaps on multiple platforms, and results must be transmitted back to the pump, all in a few seconds. Of course, other customers are likely invoking the same flow from pumps across the country at the same time.

This flow of operations crosses multiple system boundaries and combines user inputs and the results of many system service uses along the way, all to satisfy the mission objective of purchasing gasoline. In more general terms, a flow begins with a mission objective (purchase gasoline) and elaborates into a sequence of user tasks (enter data, select the product, etc.). This turns into a traversal of a complex network to locate and execute the system services (databases, business rules, etc.) required to satisfy the mission.

From an engineering viewpoint, it is easy to see that such a flow represents a specification that a system design must satisfy, and that the design must accommodate the different types and volumes of flows that its many users require. In operation, such a system must typically satisfy hundreds or thousands of such flows simultaneously. Flows must also satisfy required quality attributes such as reliability, security, and survivability. Because flows cross many security domains in multiple systems, there are many opportunities for intrusion and compromise that can impact security and survivability. If a gasoline purchase flow is compromised, it is an inconvenience. But if a flow linking sensors and weapons in a complex battle management system is compromised, it is an entirely different matter. So it is worth investigating flows and their properties to better understand security and survivability issues in complex networked systems.

Modern enterprises are irreversibly dependent on large-scale networked systems. Unfortunately, the complexity of these systems frequently exceeds current engineering capabilities for intellectual control, resulting in persistent difficulties in acquisition, development, management, and

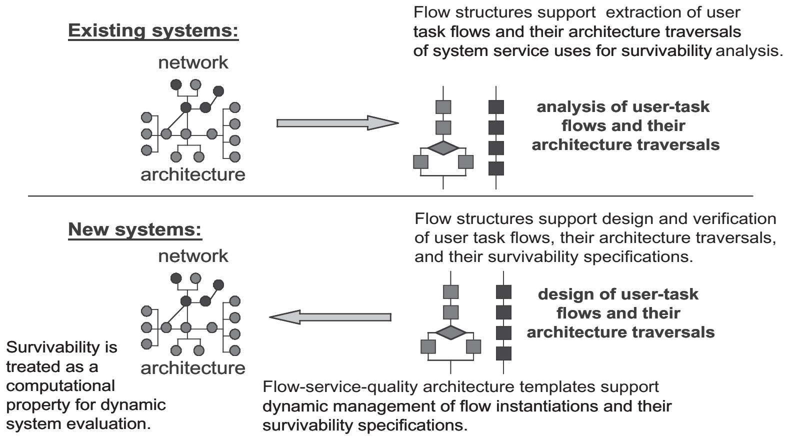

Existing systems:
network
architecture
Flow structures support extraction of user task flows and their architecture traversals of system service uses for survivability analysis.
analysis of user-task flows and their architecture traversals
New systems:
network
architecture
Survivability is treated as a computational property for dynamic system evaluation.
Flow structures support design and verification of user task flows, their architecture traversals, and their survivability specifications.
design of user-task flows and their architecture traversals
Flow-service-quality architecture templates support dynamic management of flow instantiations and their survivability specifications.

Figure 1: *Flow-Service-Quality Engineering Operations*

evolution. These systems exhibit indeterminate boundaries, ever-changing linkages to other (often stovepipe) systems, COTS capability and quality uncertainties, dynamic function and usage, and continual requirements for evolution. Complexity is compounded by extensive asynchronous behavior, that is, simultaneous shared use of system services by multiple users that results in a virtually unknowable interleaving of operations and communications among system components.

A central issue in modern system development is how to maintain intellectual control over such complex structures and the asynchronous behaviors they produce. In short, what are the stable and dependable anchors for specification and design that can provide a unified engineering discipline for large-scale network system acquisition and development? We believe that FSQ engineering can provide that discipline [5].

In complex network systems with constantly varying function and usage, flows and their corresponding architecture traversals of system services can serve as the sought-after stable foundations for functional and nonfunctional (quality attribute) specification and intellectual control. The objective of our FSQ research is to provide engineering methods to represent and reason about system flows as essential artifacts of complex system analysis and development. System flows are composed of system services and must satisfy quality attributes such as reliability, performance, and survivability. Therefore, it is these three first-class concepts, *flow*, *service*, and *quality* that form the basis of the FSQ framework for engineering large-scale network systems.

Flows can be expressed in virtually any language using flow structure templates that permit precise specification of mission objectives, corresponding user tasks, and refinements into traversals of system services. In execution, services invoked by flows typically experience a blizzard of asynchronous usage interleavings that defy human understanding. A key result of our research is an approach to flow definition that guarantees it can be expressed in simple procedural structures for straightforward human understanding and analysis, despite the underlying asynchronous behavior of its service uses.

These procedural structures embody nested and sequenced service invocations expressed in terms of ordinary sequence, alternation, iteration, and concurrent structures. Such structures enable precise refinement, abstraction, and verification of flows for human understanding. In addition, flows can be organized into related flowsets associated with particular missions and network components, and a rich set of operations can be applied to flow correlation and dependency analysis and simulation, of particular value for integrating existing stovepipe systems. Flows also define required levels of quality attributes for themselves, as well as for execution of the services they reference. FSQ engineering operations for existing and new systems are depicted in Figure 1.

In FSQ engineering, quality attributes such as security, survivability, reliability, and availability are defined as computational functions and are associated with both flows and services. Substantial effort has been devoted in the past to development of descriptive and often subjective a priori characterizations of the quality attributes of systems. Rather than focusing on descriptive predictions of limited value for dynamic networks, we adopt an alternate approach and ask how such attributes can be defined, computed, and acted upon as dynamic characteristics of system operation. That is, we wish to define quality attributes as functions to be computed, rather than as static estimates of capabilities.

While such functions rely on what can be computed and may differ thereby from traditional views of quality attributes, they can permit new approaches to attribute analysis, design, and operational evaluation. A key aspect of the computational approach is the ability to associate quality attributes with specific flows rather than with entire systems, thereby permitting differentiation among attribute capabilities based on mission criticality in survivability engineering.

In a world of flow-centric engineering and computational quality attributes, it is natural to consider system architecture templates based on dynamic flow and quality attribute management. We are investigating such FSQ architectures as straightforward implementations of flow-based systems.

Flow-structure engineering can reduce complexity and add clarity to the development of key system artifacts. First, flow specifications of enterprise tasks can be designed and verified with full human understanding (at various levels of abstraction in a rigorous and seamless process) from mission requirements down to architectural components. Second, a specification of network system behavior is defined as the set of flows of its service uses. And third, the specification of each service in a network system incorporates all its uses in all the flows wherein it appears.

Flow structures prescribe dynamic network linkages and operations, define composition requirements among nodes and services, and support both centralized and distributed control. Flow structures have the potential to reduce complexity and improve manageability in network system acquisition, development, management, and operation and can contribute to integration of diverse stovepipe systems to meet new mission requirements. In addition, flow structures can be used to extract and document mission-critical operations in existing systems to better understand component dependencies for survivability analysis.

Intrusion-Aware Design Project

Developers in many engineering disciplines rely on engineering failure data to improve their designs and methods. Imagine the result if bridge builders had ignored the lessons learned from the torsional oscillations that caused the Tacoma Narrows Bridge to collapse. Or, if ship builders had ignored the lessons learned about inadequate lifeboat space and manning that allowed the great loss of life when the Titanic sank. Engineering success requires that we also learn from the less famous disasters. The aerospace community, for example, has institutionalized a means for learning from air traffic accidents that has resulted in a very low risk of death during

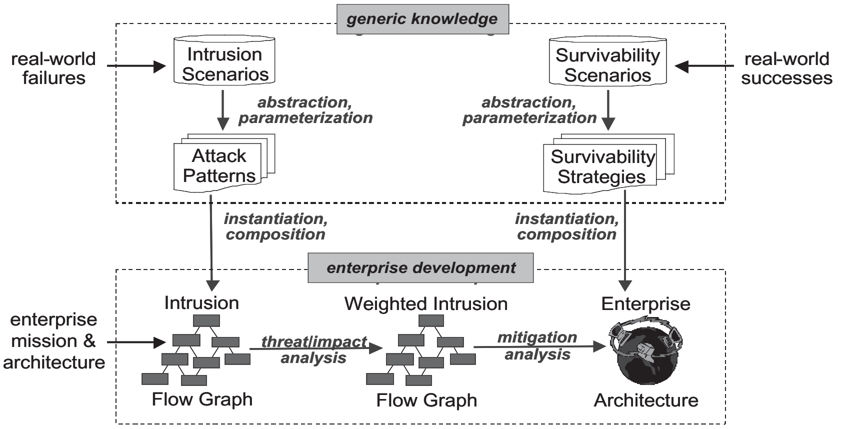


Figure 2: *Intrusion-Aware Design Refinement*

air travel, despite its inherent hazards. Successful architects design structures to survive known faults in building materials, construction methods, and the environment.

Unfortunately, information system developers generally do not use security failure or attack data to improve the security and survivability of systems that they develop. Information systems being built and managed today are prone to the same or similar vulnerabilities that have plagued them for years. In addition, increasingly sophisticated attacks exploit these vulnerabilities at an alarming rate. As seen by recent Internet worms and viruses released (e.g., Melissa, Love Letter, Code Red, Nimda), attackers share tools and knowledge to amplify their capability. The increasingly sophisticated tools currently available permit relatively inexperienced individuals to execute very sophisticated attacks. We have seen such attacks escalate with the intensity of political conflicts such as the war in Kosovo, the tensions between the United States and China, and the conflict between India and Pakistan [6]. While such attacks are often in the form of embarrassing Web site defacements, attackers are starting to target the perceptions of users, such as attempts to modify the content of major news publications or company press releases. In general, attacks can target a system's internal users and (COTS) components as well as external trusted systems and user communities.

Businesses and governments have historically been reluctant to disclose information about attacks on their systems for fear of losing public confidence or fear that other attackers would exploit the same or similar vulnerabilities. However, increased public interest and media coverage of the Internet's security problems has resulted in increased publication of attack data in books, Internet newsgroups, and CERT security advisories, for example. Much of the available attack information is very detailed in terms of software versions, enterprise-specific configurations, and attacker-specific scripts. Such details have a relatively short life as the attackers create and revise their tools and methods. However, the general patterns of attack are much less variable over time. Attack patterns describe general attack strategies, such as the various forms of denial-of-service attacks, and can be structured so they can be applied in a variety of contexts.

Intrusion-aware design methods enable information system engineers to use attack patterns in a structured way to improve information system security and survivability. Our approach is to collect as much knowledge about attack patterns and survivability strategies as possible to support the development and analysis of specific enterprises. Such a knowledge base can assist in identifying the general system risks and the most appropriate mitigation techniques. For example, network-based denial-of-service attacks suggest the need to distribute and diversify critical services, provide spare capacity, and/or attempt intruder trace-back, filtering, and possible apprehension.

Attack and survivability information needs to be structured and reusable so they can be applied in the iterative refinement of survivability architectures. By building the knowledge base so that it is independent of the enterprise, we provide a means for building enterprise-specific intrusion flow graphs in an affordable way, thus making the iterative refinement and analysis of the enterprise architecture cost-effective.

As shown in Figure 2, we build intrusion scenarios from real-world failures documented in, for example, incident and vulnerability databases. This effort requires fusing sometimes low-level incident data together to understand and describe larger-scale intrusions. We interpret intrusions broadly to include attacks that target people and task flows as well as those that target technology. We develop a means to derive commonly recurring attack patterns from intrusion scenarios. These attack patterns are parameterized so that they can be instantiated for varying enterprise environments. Enterprise-specific intrusion flow graphs are generated from these attack patterns through an instantiation and composition process [7].

Risk analysis techniques are used to prioritize the intrusions through threat and impact analyses. Mitigation analysis of these intrusions helps identify relevant survivability strategies that are used to refine the enterprise architecture in the most beneficial directions. The survivability strategies are derived from real-world survivability scenarios documented through years of practical experience in the area.

Intrusion-aware design does not reinvent risk analysis, but uses and augments risk analysis and management techniques where helpful. Our near-term focus is to explore the viability of this approach through its application to improve security and survivability of a particular enterprise architecture for a particular class of attacks. With evidence of the method's efficacy, our efforts will shift to developing and structuring the generic knowledge for intrusion and survivability scenario analysis (top part of Figure 2). Showing how to use this generic knowledge with existing risk management techniques for intrusion analysis and architecture improvement (bottom part Figure 2) is also a focus and key to the success of the approach. The key benefits of the approach are as follows:

- More structured/systematic means to document enterprise threats.
- Better understanding of enterprise mission vulnerability to sophisticated, multi-stage attacks.
- Improved accuracy and speed of risk analysis and management activities.
- Improved ability to identify architectural strategies to counter likely, high-consequence attacks.
- Faster, iterated improvement to enterprise architecture and overall survivability.

Successful application of intrusion-aware design methods should lead to enterprise architectures that demonstrably tolerate sophisticated attacks, providing higher confidence that the enterprise successfully carries out its mission.◆

Acknowledgements

It is a pleasure to acknowledge contributions to this work by Dr. Alan Hevner, Dr. Mark Pleszkoch, Dr. Carol Sledge, Dr. Ann Sobel, and Dr. Gwen Walton.

References

1. Anderson, R. H., A. C. Hearn, and R. O. Hundley. "RAND Studies of Cyberspace Security Issues and the Concept of a U.S. Minimum Essential Information Infrastructure." Proceedings of the 1997 Information Survivability Workshop. IEEE Computer Society, San Diego, Calif., 12-13 Feb. 1997. Available at <www.cert.org/research/isw/isw97/front_page.html>.
2. Ellison, R., D. Fisher, R. C. Linger, H. F. Lipson, T. Longstaff, and N. R. Mead. Survivable Network Systems: An Emerging Discipline (CMU/SEI-97-TR-013, ADA341963). Pittsburgh, Penn.: Software Engineering Institute, Carnegie Mellon University, Nov. 1997, revised May 1999. Available at <http://www.cert.org/research>.
3. Mead, N. R., R. J. Ellison, R. C. Linger, T. A. Longstaff, and J. McHugh. Survivable Network Analysis Method (CMU/SEI-2000-TR-013). Pittsburgh, Penn.: Software Engineering Institute, Carnegie Mellon University, 2000. Available at <www.sei.cmu.edu/publications/documents/00.reports/00tr013.html>.
4. Lipson, H. F., N. R. Mead, and A. P. Moore. Can We Ever Build Survivable Systems From COTS Components? (CMU/SEI-2001-TN-030). Pittsburg, Penn.: Software Engineering Institute, Carnegie Mellon University, 2001. Available at <www.sei.cmu.edu/publications/documents/01.reports/01tn030.html>.
5. Hevner, A. R., R. C. Linger, G. Walton, and A. Sobel. "The Flow-Service-Quality Framework: Unified Engineering for Large-Scale Adaptive Systems." Proceedings of the Hawaii International Conference on System Sciences-35. Los Alamitos, Calif.: IEEE Computer Society Press, 2002.
6. Vatis, M. A. "Cyber Attacks During the War on Terrorism: A Predictive Analysis." Institute for Security Technology Studies at Dartmouth College, 22 Sept. 2001.
7. Moore, A. P., R. J. Ellison, and R. C. Linger. Attack Modeling for Information Security and Survivability (CMU/SEI-2001-TN-001, ADA388771). Pittsburg, Penn.: Software Engineering Institute, Carnegie Mellon University, 2001. Available at <www.sei.cmu.edu/publications/documents/01.reports/01tn001.html>.

About the Authors

Robert Ellison, Ph.D., is a senior member of the technical staff in the Software Engineering Institute's Networked Systems Survivability Program. Dr. Ellison's research interests include system survivability and architectural patterns and styles for security architectures. He has a doctorate in mathematics from Purdue University and is a member of the Association for Computing Machinery and International Electrical and Electronics Engineers Computer Society.

Richard Linger is a senior member of the technical staff at the Software Engineering Institute's CERT Coordination Center at Carnegie Mellon University (CMU). He teaches at the CMU H. J. Heinz School of Public Policy and Management. Linger has published three software engineering textbooks and more than 50 articles. He has a bachelor's degree in electrical engineering from Duke University. He is member of the International Electrical and Electronics Engineers Computer Society and the National Software Council.

Howard Lipson, Ph.D., has been a computer security researcher at the Software Engineering Institute's (SEI) CERT Coordination Center for 10 years. Dr. Lipson played a major role in extending security research at the SEI into the new realm of survivability, developing many foundational concepts and definitions. Dr. Lipson has been a chair of three International Electrical and Electronics Engineers Information Survivability Workshops. He has a doctorate degree in computer science from Columbia University.

Nancy Mead, Ph.D., is the team leader for the Survivable Systems Engineering team as well as a senior member of the technical staff in the Networked Systems Survivability Program of the Software Engineering Institute (SEI), and a faculty member in the Masters of Software Engineering program at Carnegie Mellon University. She is currently involved in the study of survivable systems requirements and architectures, and the development of professional infrastructure for software engineers. Prior to joining the SEI, Dr. Mead was a senior technical staff member at IBM Federal Systems. She also worked in IBM's software engineering technology area and managed IBM Federal Systems' software engineering education department. She has developed and taught numerous courses on software engineering topics, both at universities and in professional education courses.

Andrew Moore is a member of the technical staff at the CERT Coordination Center of Carnegie Mellon University's Software Engineering Institute (SEI). As a participant in the Network Systems Survivability Program, Moore explores ways to improve the survivability of unbounded systems. Before joining the SEI, he worked for the Naval Research Laboratory investigating high-assurance system development methods for the Navy. His research interests include survivable systems engineering, formal assurance techniques, adversary modeling, and security risk analysis. Moore has a bachelor's of arts degree in mathematics from the College of Wooster and a master's of arts degree in computer science from Duke University.

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburg, PA 15213-3890

Joe Jarzombek, deputy director of Software Intensive Systems OSD (AT&L), began the award presentation ceremony for winners of the 2001 Top 5 U.S. Government Quality Software Projects (see page 30 for full coverage).

Members of the co-sponsors panel pose prior to taking questions from attendees.

The conference provided a forum for audience participation through questions.

STC 2002

Onlookers cheered and lamented as judges recorded flight distances at the second annual CrossTalk *Paper Airplane Contest held during STC 2002.*

14th Annual Software Technology Conference Hosts Thousands

One of the largest co-sponsored events for U.S. defense-related software technologies, policies, and practices drew more than 2,100 attendees from around the world recently to Salt Lake City. The Software Technology Conference (STC 2002) tackled the theme, “Forging the Future of Defense Through Technology” from April 29-May 2 at the Salt Palace Convention Center.

“The U.S. Department of Defense (DoD) must make a fundamental shift to network-centric warfare away from platform-centric warfare … to accommodate a global information grid in a nontraditional and asymmetrical threat environment,” said Dr. Margaret E. Myers, principal director, deputy assistant secretary of defense, in her opening keynote speech. “We can’t keep the bad guys out. So we have to find another way to protect ourselves,” said Myers. Like the human body with its internal barriers to germs and viruses, the U.S. defense department must have a network that provides information dominance in a secure environment, she said.

Stephen L. Squires, vice president and chief science officer at Hewlett-Packard (HP) Co., said in his talk that “the single most important thing we have in the DoD, in addition to the leaders, are the kids in the trenches who know the technology. Information technology became a munitions after Sept. 11.” Squires told how private industry, including HP, AT&T, and more, “took all their toys” to build a command center in Florida after the attack. “Kids in blue jeans worked with the military,” he said.

Myers and Squires were just two of the many industry, government, and defense leaders who spoke at the conference. The 14th annual STC featured 164 different exhibitors and 160 speaker presentations, ranging from software systems/architecture, maturity models, common and open systems to information assurance, best practices and more. In all, speakers pointed toward a future that will demand increased quality, best value, and on-time delivery for the government’s future software projects in an environment that demands information assurance to best support customers – the war fighters.

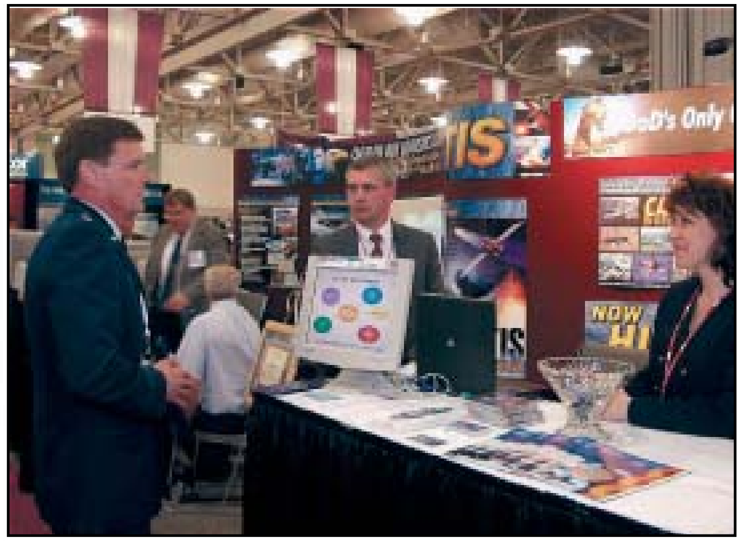
In addition to the educational and training opportunities at the STC 2002, the conference gave attendees a chance to network at all levels at a variety of planned events throughout the week.

On Tuesday afternoon, the second annual CrossTalk Paper Airplane Contest entertained participants and viewers alike. More than 100 entrants flew individually designed airplanes in a fly-off from the Salt Palace foyer. The paper airplane contest is co-sponsored by CrossTalk, *The Journal of Defense Software Engineering*, and Shim Enterprise, Inc.

The STC is co-sponsored by the U.S. Army, Marine Corps, Navy, Air Force, the Defense Information Systems Agency (DISA), and Utah State University Extension. Representing the services in a panel held Tuesday morning were (pictured in middle photo at left) LTG Peter M. Cuviello, U.S. Army; Debra M. Filippi, U.S. Marine Corps; RADM Tom S. Fellin, U.S. Navy; Diann L. McCoy, DISA; John M. Gilligan, U.S. Air Force; and moderator Dawn C. Meyerriecks, also of DISA.

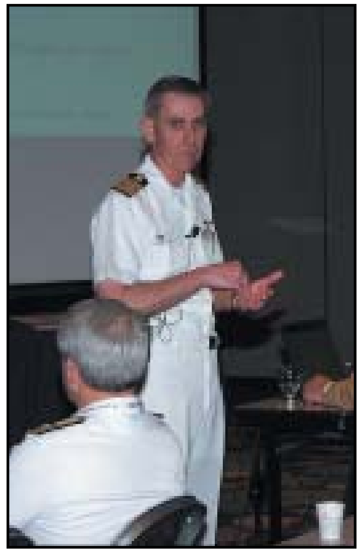
The co-sponsors have already started planning STC 2003, scheduled for April 28-May 1, 2003.◆

Photos by Randy Schreifels of the Software Technology Support Center and Janna Kay Jensen of CrossTalk.



Maj. Gen. Scott Bergren, commander Ogden Air Logistics Center, visits the Ogden Air Logistic Center’s Technology and Industrial directorate Software booth.

Dr. Margaret E. Myers, principal director, deputy assistant secretary of defense, gave the keynote opening presentation on Monday morning.



Individual service sessions like this one for the U.S. Navy allowed attendees to hear from leaders of all four major branches of the military.

There was plenty of room for listening and hands-on learning at one of 160 daily presentations.

Security Issues with SOAP

Jim Clune and Dr. Adam Kolawa
ParaSoft Corporation

Formerly known as Simple Object Access Protocol, SOAP is rapidly becoming the standard for building Web services and connecting disparate systems in a loosely coupled fashion with complete platform independence. However, some of the very features that make SOAP attractive, such as its flexibility and its compatibility with HTTP, also provide opportunities for security breaches. This article discusses SOAP security issues and how they can be addressed.

SOAP (formerly known as Simple Object Access Protocol) is a lightweight protocol for exchanging structured and typed information in a decentralized, distributed environment. It is an XML-based protocol that consists of three parts:

- An envelope that defines a framework for describing what is in a message and how to process it.
- A set of encoding rules for expressing instances of application-defined data types.
- A convention for representing remote procedure calls and responses.

SOAP makes possible a universal platform for Web-based applications that transcend the boundaries of a specific programming language and/or specific platform. With SOAP users/adaptors growing by the day, SOAP is rapidly becoming the standard for building Web services and connecting disparate systems in a loosely coupled fashion with complete platform independence.

SOAP was developed by Microsoft, DevelopMentor, and Userland Software and proposed as an XML protocol to the World Wide Web Consortium (W3C). The name reflected the idea that SOAP would be used to express serialized object graphs, enabling object-oriented systems to perform functions such as remote procedure calls while preserving objects and their relations. However, in the W3C's latest working draft (version 1.2), SOAP became the name and is no longer an acronym. This reflects a shift in thinking about SOAP from a serialization framework for object-oriented systems to a more general XML-based messaging paradigm, where the messages do not necessarily contain objects.

SOAP is a key technology enabling the development of *Web services*, which is a term that has emerged to describe a software module deployed on the Web and intended for use as a component in one or more applications distributed across the Internet. The promise of Web services is to facilitate the creation of open distributed systems that leverage networks by aggregating multiple services and providing higher levels of functionality.

Unlike the distributed computing protocols that preceded it, SOAP is simpler, more flexible, and facilitates looser coupling between the components. For example, the Object Management Group's Internet Inter-Object Request Broker Protocol (IIOP) is the underlying transport mechanism used by the Common Object Resource Broker Architecture (CORBA). Microsoft's Distributed Component Object Model (DCOM) is a distributed computing protocol that extends Component Object Model (COM). Whereas IIOP is tightly coupled to CORBA's heavyweight architecture and infrastructure, and DCOM is tied into Microsoft's COM architecture, SOAP is not tied to any corresponding architecture or infrastructure.

> *"Depending on how your software is configured, a remote operator could access your system and provide instructions to your server."*

SOAP is a stateless, one-way messaging paradigm. Although more complex interaction patterns can be built on top of SOAP, the protocol is not tied to objects or an infrastructure managing them. SOAP is built on XML, which lends itself to cross-platform interoperability. For the transport layer, SOAP commonly uses HTTP, another text-based communication protocol that has gained wide acceptance, as is evident from the state of the Web today.

However, the very attributes that make SOAP so attractive also give cause for some concern. The ports that serve as integration points for business partners can serve as entry points for unwanted elements, such as hackers and viruses. Depending on how your software is configured, a remote operator could access your system and provide instructions to your server. This is a security issue. SOAP's openness and flexibility, the very things that make it so powerful, can enable attackers to wreak havoc on your system.

How can you protect yourself? The rest of this article explores and addresses the security issues that are relevant to existing technologies. Some of these challenges are applicable to a number of existing protocols, but for the scope of this article, we are focused specifically on SOAP. After addressing these issues, a discussion of the challenges beyond the mainstream solutions will follow, as well as ideas for meeting these challenges.

Security Issues and Priorities

Security is not a single problem, but rather a host of interrelated issues. For any given application, some of the issues will be critical, while others may be of lower priority or even irrelevant. Here are some facets of security that are worth considering when deploying SOAP services:

- Privacy: For many services it is important that messages are not visible to anyone except the two parties involved. This means traffic will need to be encrypted so that machines in the middle cannot read the messages.
- Message Integrity: This provides assurance that the message received has not been tampered with during transit.
- Authentication: This provides assurance that the message actually originated at the source from which it claims to have originated. You may need to not only authenticate a message, but also prove the message origin to others. This is called non-repudiation. Non-repudiation is a legal

concern as well as a technical issue, and achieving it is beyond the scope of this article.

- Authorization: Clients should only be allowed to access services they are authorized to access. Note that authorization requires authentication, because without authentication, hostile parties can masquerade as users with the desired access.

The first step in implementing security is to determine which aspects are important for your organization. It is also helpful to have some idea of their priorities with respect to each other as well as in terms of non-security related goals such as quality and performance.

Implementing Security with Current Technology

SOAP is a young technology, so the tools and techniques for using it are still evolving. However, engineers under a deadline need to understand what is available *now*.

In order to achieve security goals, we will first look at HTTPS, which is HTTP over the secure socket layer (SSL). The SSL is widely used on the Internet today. It performs public key encryption to address the privacy aspect of security. It also performs a message integrity check using a keyed message authentication code.

When it comes to authentication, the story gets a little more complicated. SSL uses certificate-based authentication, and the certificates for the server and the client may be controlled independently. By far, the most common usage is for the server to have a certificate and the client not to have a certificate. In this scenario, the client has assurance of the server's identity, but the server does not have assurance of the client's identity. However, SSL may also be configured so that both the server and the client require trusted certificates.

Certificates for SSL are often, but not required to be, in a chain with the root certificate from a well-known trusted authority such as Verisign, Thawte, Entrust, etc. HTTPS can be used in conjunction with other types of authentication such as HTTP Basic and Digest authentication. By itself, Basic authentication is an extremely weak protection scheme since it involves sending username and password in base64 encoded plain text. For some applications this is sufficient. Digest is considerably better because it involves a challenge/response mechanism where the password is not sent directly. Combining either of these with HTTPS makes for a significantly more secure connection since encryption provided by the SSL prevents hackers from spying the passwords and reusing them.

Once you have established a means of authentication, you need to establish authorization procedures. It is helpful to think of authorization in two broad categories: *declarative* and *programmatic*. Declarative authorization typically involves specifying which groups various users belong to and which groups can access each service. Here the membership of each group determines the complete specification, so coding is either trivial or nonexistent. Programmatic authorization involves obtaining the user, group, or role at runtime and using that information to perform some logic about what to do next. Programmatic authorization is more flexible in that you have more options about what criteria to use to reject a request as well as what action to take if a request comes from a non-trusted party. The trade-off is that programmatic authorization is more complex, involves writing code, and provides more opportunities for error.

> *"Breaches in security are often the result of false assumptions. The most dangerous are the ones that are implicit and unspoken."*

Example: A Financial Institution

Although SOAP-based Web services can be deployed in a wide variety of languages and platforms, the principles used are best illustrated by a concrete example. In our discussion, we will be developing our services in Java, utilizing the Apache SOAP implementation, and deploying them over HTTP using the Apache Tomcat servlet container. Each of these tools is freely available for commercial use.

We will use the example of a financial institution using SOAP-Remote Procedure Call (RPC) for business-to-business integration. Before we jump into the security issues, we will briefly review the relevant pieces for implementing this scenario. A simple method that a financial institution may want to expose is: getAccountBalance(account Number), which takes an account number as an input and returns the current account balance. The Java method signature may look like this:

```
public int getAccountBalance
(int accountNumber) throws
InvalidAccountException {
// Perform database query and
return result.
...
}
```

This describes the interface for a Java application, but for SOAP we need to translate this into appropriate SOAP terms. The current approach is to create a Web Service Description Language (WSDL) document. WSDL is another XML-based language that has been proposed to the W3C. It is used for describing services as a set of endpoints operating on messages. This WSDL then becomes the published interface for business partners utilizing the service. Deploying the implementation requires configuring our components (in our case, Tomcat, Apache SOAP, and our implementation Java class working together).

Deploying the service over HTTPS requires only minor modifications to the HTTP. First, you will need a certificate for the server. This certificate is the identifier that enables clients to authenticate the server. Java manages private keys and their associated certificates in keystores. Conceptually, keystores are databases of key entries and trusted certificate entries, though they are often implemented in files rather than relational databases. Java also provides a tool for managing keystores called keytool. Keytool generates self-signed certificates as well as certificate signing requests, which are sent to a certificate authority.

Next, both the interface and the implementation must be modified to reflect the change in protocol. For the interface, change the WSDL to specify the HTTPS protocol in the RPC router, which indicates where to route the remote procedure call. For the implementation, enable an HTTPS connector in the Tomcat server configuration file.

The client will also need to make some minor changes. The URL changes to reflect the use of HTTPS instead of HTTP. (This will happen automatically if the client is WSDL-aware.) For accepting certificates, Java again uses a keystore. The keystore should contain the trusted certificate entry corresponding to the certificate associated with the server. This assures the client that the response really comes from our financial institution and not a malicious party intercepting the request. The keystore can be configured statically using keytool or it can be accessed and manipulated programmatically at runtime in the client.

Beyond the Fundamentals: Everything You Know Is Wrong

Breaches in security are often the result of false assumptions. The most dangerous assumptions are the ones that are implicit and unspoken because these are made subconsciously, so they are never directly challenged and scrutinized. Here we present some ideas to challenge the reader's assumptions. In some respects these ideas are simply common sense. However, in the words of the French philosopher Voltaire, "Common sense is not so common."

There Is No Guaranteed Security

Public key cryptography systems are a very good technology, but they are not a panacea. What takes 50 years to break using brute force on today's most powerful supercomputer may take three seconds after an unexpected breakthrough in quantum computing. More pointedly, it does not matter how long brute force takes if the hacker does not use brute force. The easiest way to circumvent public key encryption is to gain physical access to a computer containing the private key. When someone tries to tell you security is guaranteed, remember that whenever humans are involved, guarantees are an illusion, which only serves to prevent you from thinking about what can go wrong.

Security Through Obscurity Is Not Always Bad

The term *security through obscurity* is used to describe security measures that rely on secrets in the protocol or algorithm. This is in contrast to public key cryptography systems, which have secret (private) keys, but well-known algorithms. Despite the inherent weaker nature of the security through obscurity approach, if it is layered on top of strong encryption schemes, it can provide an additional deterrent for would-be attackers. In addition, this hybrid approach has an advantage over a strict, strong cryptography scheme since in a traditional scheme the attacker knows that all he needs is the private key. In an obfuscated scheme, however, he may not have any clue what parameters are relevant.

How does this apply to SOAP messages? SOAP is a very expressive protocol. There are in fact an infinite number of variations on how to say the same thing. For example, white space in certain contexts in XML documents is specifically defined as ignorable. Yet there is nothing to prevent you from requiring specific, obscure rules for white space in your SOAP messages. Attackers familiar with SOAP would assume that the ignorable white space is really ignorable, giving you the opportunity to turn the tables and capitalize on the hackers' assumptions. Another example would be to put constraints on the namespace prefixes used, another area where multiple solutions are possible. Outside the SOAP envelope, the HTTP header could also house additional constraints. The obvious side effect is that we have completely undermined the interoperability of SOAP.

> *"Whenever humans are involved, guarantees are an illusion, which only serves to prevent you from thinking about what can go wrong."*

Interoperability Is Not Always Good

As mentioned in the beginning of this article, a major benefit of SOAP compared with other distributed computing technologies is the evidence that the promise of cross-platform and cross-language interoperability is finally being realized. However, security requirements like authentication and authorization can be reformulated as requirements to prevent interoperability with malicious parties. If you already know what implementation your intended clients are using, then being able to interoperate with other clients may be an asset or a liability, or both.

It Is All About Patterns

Another way to look at the security problem is to recognize that the whole process of deploying Web services, as well as the mechanisms of invoking SOAP RPCs and processing SOAP messages, is about manipulating structured data. Within this data are many patterns, some of which promote security, while others undermine security. In this sense, many security problems can be reformulated as a requirement to identify insecure patterns and prohibit them from infecting the system. Patterns in XML are especially important to SOAP because XML permeates the entire architecture. XML may appear in the following:

- Server configuration files.
- Deployment descriptors.
- WSDL.
- In the SOAP envelope.

When a security hole is introduced in any of these components, there is an XML pattern that corresponds to that security hole. For example, your server configuration file could be set to not only expose your service over HTTPS on port 443, but it could expose the same servlet over HTTP on port 80. In some cases, this is desirable, but only if you intentionally make the service available through both a secure and a nonsecure connection. Finding these types of patterns in XML is another technique to ensure security.

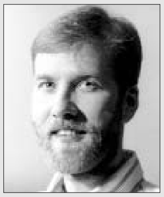
It Is All About Layers

Providing security in multiple layers increases robustness. If one security layer is compromised, the next security layer still provides protection. Layers can also be used to provide flexibility by encrypting some parts of an XML document differently than others. This allows users access to only the portions of the document related to them. Work in the area of encrypting parts of XML documents is under way in the XML Encryption working group of the W3C. A related area is the idea of providing means for signing XML data and verifying signatures, which is covered by the XML Signature working group of the W3C.

Conclusions

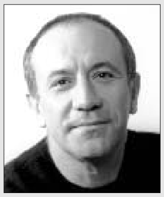
The potential exists for SOAP to allow you to set up very dynamic Web services highly customized to the specific needs of each of your customers. However, this new opportunity comes with the challenge of being able to consistently provide flexibility without compromising your security. Meeting the challenges of security needs requires knowing what the security needs and priorities are, what technologies can be used to achieve them, and above all, thinking clearly about your system's weaknesses.◆

About the Authors



Jim Clune is development manager for ParaSoft Corporation as well as technical lead for the SOAPtest development team. His professional experience includes software engineering, manufacturing engineering, and management. He has a master's of science degree in applied computer science and technology from Azusa Pacific University and a bachelor's of science degree in engineering from Harvey Mudd College.

ParaSoft Corporation
2031 South Myrtle Avenue
Monrovia, CA 91016
Phone: (888) 305-0041
Fax: (626) 305-3036
E-mail: jim.clune@parasoft.com



Adam Kolawa, Ph.D., is CEO and founding member of ParaSoft Corporation. Dr. Kolawa's experience with various software development processes allows him insight into the high-tech industry providing him the ability to successfully identify technology trends. Dr. Kolawa holds seven patents for the technologies behind several innovative commercial software tools. He is co-author of "Bulletproofing Web Applications" and was awarded the Los Angeles Ernst & Young's Entrepreneur of the Year Award in the software category. Dr. Kolawa has a doctorate degree in theoretical physics from the California Institute of Technology.

ParaSoft Corporation
2031 South Myrtle Avenue
Monrovia, CA 91016
Phone: (888) 305-0041
Fax: (626) 305-3036
E-mail: ukola@parasoft.com

WEB SITES

Defense Information Systems Agency

www.disa.mil

The Defense Information Systems Agency (DISA) is a combat support agency responsible for planning, developing, fielding, operating, and supporting command, control, communications, and information systems that serve the needs of the Department of Defense (DoD) and other government offices. DISA is a provider of integrated information solutions to DOD and non-DOD customers.

National Communications System

www.ncs.gov/n5_hp/n5_ia_hp/default.htm

The National Communications System (NCS) Information Assurance Branch was established to focus on the network and information security initiatives of the NCS under a common program branch. This was done to increase NCS efficiency and effectiveness, apply a coordinated direction, and increase the general awareness of the importance of network and information security to the NCS government and industry community. This site is a link to the NCS home page listing programs, publications, member organizations, and more.

Software Testing Institute

www.softwaretestinginstitute.com

The Software Testing Institute (STI) provides access to quality industry publications, research, and online services. STI offers the following professional resources: a software testing discussion forum, the STI Resource guide, the Automated Testing Handbook, the STI Buyer's Guide, and privileged access to STI's exclusive industry surveys, including salary and staffing practices, industry trends and more.

World Wide Web Consortium

www.w3.org

The World Wide Web Consortium (W3C) develops interoperable technologies to lead the Web to its full potential as a forum for information, commerce, communication, and collective understanding. On this page, you'll find W3C news as well as links to information about W3C technologies, including Simple Object Access Protocol (SOAP) 1.1, XML Encryption WG, W3C architecture domain, industry surveys on salaries, staffing practices, industry trends, and more.

Focusing Measurement on Managers' Informational Needs

Peter Baxter
Distributive Software

Establishing a measurement process has evolved from the days of, "If it moves, count it," through the goal-question-metric period to today's information-needs-based approach for identifying and defining what to measure. Measurement process guidance from ISO and Practical Software and Systems Measurement provides a robust and flexible framework for measurement, but they only identify the purpose of measurement as the "informational needs" of managers. What are these informational needs? This article describes simple techniques for identifying informational needs within your organization, i.e., informational needs that become the requirements of your measurement process and lead to a useful and effective measurement process.

In many organizations, a measurement process is a required element in managing technology programs. To meet this requirement, several groups have initiated projects to develop and standardize a set of *best practices* for setting up such a measurement process. Three principle sources of measurement guidance that have converged from these groups' efforts during the past two years are as follows:

- The emerging ISO standard, ISO/IEC 15939 Software Measurement Process.
- The Practical Software and Systems Measurement (PSM) Guidebook version 4 [1].
- The Software Engineering Institute's Capability Maturity Model® (CMM®) Integration^{SM} (CMMI^{SM}) project [2].

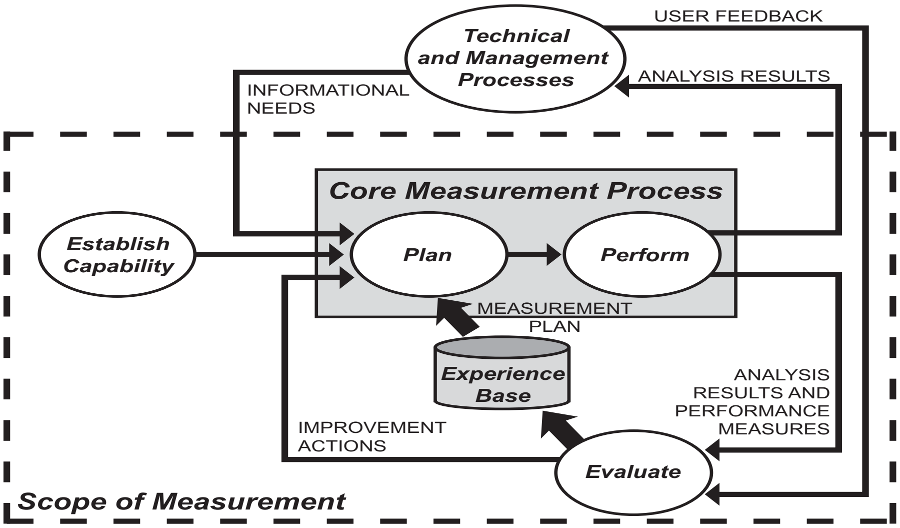
Measurement guidance and principles are consistent among these three documents, with the basic measurement process model shown in Figure 1.

As shown in Figure 1, *informational needs* drive the planning, performance, and evaluation activities within the measurement process. Informational needs are the requirements of essential measurement activities in the *core measurement process*. Once informational needs are defined, a *measurement plan* is developed by decomposing them into *analysis results and performance measures* (information products), which contain measures and associated guidance. These information products are delivered to managers and drive *improvement actions*.

By placing informational needs outside the scope of the core measurement process itself, the measurement process can be used to support a wide range of management and executive functions. While this makes measurement a flexible process, it also requires that informational needs be clearly reviewed, prioritized, and documented *before* initiating or expanding a measurement process. Because identifying informational needs is a critical step in ensuring measurement process success, organizations must take the time to correctly select and define these needs.

This article presents common types of informational needs that are found in both commercial and government organizations involved in systems and software engineering. Many also address the informational needs of key practices from the CMMI. These common needs should encourage you to extract potential measurement process requirements from your organization's management, system or software, and support processes. The following nine techniques provide tools for extracting the real informational needs within your organization.

Figure 1: *Measurement Process Model*



1. Identify Informational Needs for Current Management Practices

Organizations often develop a culture that encompasses technical management and engineering functions. This culture develops simultaneously as process, training, infrastructure, and habit evolve. For example, in a systems acquisition program, a monthly meeting may be held to review the supplier's progress and address potential risks. This meeting is a result of the culture and process within the organization, which has found that a periodic review of progress has led to a greater probability of on-time delivery.

From a measurement perspective, the periodic meetings represent a set of candidate informational needs that, when satisfied, makes the culture more effective and efficient. The information needed at the periodic meetings becomes a formal *information need* of the measurement process.

One of the primary barriers to measurement adoption is the misalignment between how managers work and what information the measurement process

® Capability Maturity Model, CMM, Software Capability Maturity Model, and SW-CMM are registered in the U.S. Patent and Trademark Office.

SM CMM Integration and CMMI are service marks of Carnegie Mellon University.

provides. By examining how managers really work, and using that to drive the measurement process, you are more likely to achieve greater measurement process adoption and success.

2. Identify Requirements' Measurements

Without exception, systems and software managers must measure the requirements engineering activities of the life cycle. Measuring requirements engineering activities involves quantifying the progression of software requirements from concept to formulation to design to test. Assessing these requirements ensures that your product contains all required functionality.

Typically, program plans and projections are based on estimates of software requirements, which are used as the basis for software size estimates. Because estimating requirements plays such a large part in developing the initial program plan, it is imperative to monitor that requirements are proceeding as expected. Consider a scenario where you are developing 25 percent more requirements than you planned – every life cycle activity may then be 25 percent or more over schedule and budget.

It is advisable to measure the number of requirements that each software process generates or accepts. Measure the number of system or top-level software requirements (i.e., *features* or *capabilities*), as well as the decomposition of system requirements into more detailed requirements. Measuring requirements helps you to keep tabs on the scope of your program. One of the most common issues detected by measuring requirements is *requirements creep*: the tendency to keep adding requirements to a program without considering how many additional resources or risks those new requirements represent.

In order to track differences between developed and planned requirements, it is necessary to also measure the status of each requirement as it moves through life cycle activities. A typical requirement status could be as follows: defined, approved, allocated, designed, implemented, tested, and verified. For example, in the CMMI requirements management process area, one of the typical work products identified for sub-process 1.3 Manage Requirements Changes is *requirements status*. A practical sample of how to define requirements status and then manage status changes can be found in the "CMM Implementation Guide" [3].

A measure that shows the status of all requirements is essential to monitoring program status and acts as a scorecard to illustrate that requirements are being implemented. Early in the program schedule, ensure that requirements become defined, approved, and allocated as the system architecture is finalized. Near the end of the program schedule, you should see requirements move from implemented status to tested then to verified status. While valuable in detecting *requirements volatility*, this measure also supports monitoring effort, configuration management, and quality.

3. Identify Risks That Impacted Previous Programs

In most organizations, as well as in the experience of many managers, there is a history of project lessons that should not be repeated! This includes reasons that projects were never completed, or why projects were delivered late, over budget, and without needed functionality. These lessons learned in similar and recent software programs are prime candidates for measurement in the next (now current) program.

> *"By placing informational needs outside the scope of the core measurement process itself, the process can be used to support a wide range of management and executive decisions."*

With historical risks, focus on identifying the cause of the risk rather than the symptom or the response. For example, with a project where the software was delivered late, try to remember and uncover the specific software components that led to the lateness. Perhaps the reason for the program delay was that a key piece of commercial off-the-shelf (COTS) software was not available or that the integration took longer than expected.

Consider also that software managers are often aware of software problems but only react when a schedule delay is required. In our own software development, we have been *burned* by technical and schedule problems related to our COTS vendors, and we have essentially let their problems impact our projects. We now take an *act early* approach where our product manager immediately considers technical alternatives once a problem is identified. In your environment, consider whether you want to measure to *detect* technical problems for monitoring, or to *take management action* on known ones. In some cases, your measurement program will need to do both.

4. Identify Risks for the Current Program

During program planning, you may establish a risk management plan. For each risk, you typically develop risk identification, mitigation, impact, and probability. A measurement process can support a risk management plan by identifying risks that need to be mitigated and by quantifying the effect of mitigation activities. From a measurement perspective, risks can become informational needs that drive the measurement process. The measurement process can, and should, address these needs.

Since there are costs associated with measurement (as well as risk management), you may want to select a subset of risks to measure. You could use probability and estimated mitigation cost (or impact) as a discriminator in selecting risks to measure. For example, you may choose to measure only high- and medium-probability risks where the associated mitigation cost is greater than $100,000.

A common risk within our organization is that software developers will not spend as much time as planned on a given product baseline. In the past, we found that senior developers were temporarily *borrowed* for other product development or support activities, for example, to investigate the cause of a field report from a customer. To address this risk, we developed an information need related to ensuring that resource expenditures correspond to our business priorities, i.e., first things first.

5. Measure What You Are Trying to Improve

It is the author's experience to frequently see organizations implementing large-scale, institutional change without implementing the corresponding means for managing the resultant process. Tom DeMarco, consultant and author, coined a phrase commonly heard in the world of software management: "If you don't measure it, you can't manage it" [4]. So, before you take a small step to improve an isolated software task, or a large step to improve an entire process, consider how you will demonstrate actual process improvement.

In addition to desiring to better manage your new or changed software process(es), be aware that there is an even more important reason to measure the processes that you are attempting to improve – to develop a quantitative understanding of why your software process behaves as it does. Developing this type of quantitative process understanding requires being able to mathematically describe the primary process factors. Once this mathematical relationship is established, the next step is to monitor and control the effects of these process factors. Furthermore, your estimates will become more accurate as a result of this better understanding.

Many measurement practitioners confuse a general quantitative understanding of their process with the quantitative management capability included in the CMM Level 4 Optimized. In practice though, organizations develop quantitative models for activities ranging from requirements engineering, inspections, defect detection and removal, to system testing software release activities – and few of them are rated at CMM Level 4. The point here is that by developing an understanding of your processes through measurement, you will be in a better position to estimate, control, and manage them, and less likely to rely on subjective guessing. (In other words, do not wait until you are attempting a CMM Level 4 assessment to start measuring; start now and you will be that much ahead of the game.)

6. Identify Software Quality Measures

Some years ago, a former market-leading technology company decided to counter its market slump by hiring a technology vice president. At the first meeting of his direct reports, he walked around the table, put an airsickness bag in front of each person and said, "Your schedules make me sick." He went on to say that schedules without quality do not mean anything.

In essence, it does not matter how well you stick to the schedule if the system or software product is unusable by the customer. This vice president knew what the market demands – it is unfortunate that more companies do not take quality seriously. If they did, they would focus on building a quality product rather than racing to get a substandard product to market quickly.

System or software quality is more than measuring the quality of the end product. End product quality is the result of the systems and software quality activities employed during development. If you ignore the quality aspects of systems and software development, it is anybody's guess what the quality of the end product will be.

> *"There is an even more important reason to measure the processes that you are attempting to improve – to develop a quantitative understanding of why your software process behaves as it does."*

One technique for addressing software quality is to use *quality gates*. This involves establishing reasonable and measurable thresholds at several points during development and then ensuring that the software or work products meet them before continuing. A quality gate could be all requirements approved, all unit tests passed, all code inspected, or all requirements (or subset) tested. Microsoft, for example, required developers to have no showstoppers or priority one bugs in their code in order to release Windows NT to beta testing [5]. Microsoft managers plan on several zero defect releases during the development of a product. You should use the appropriate measures to determine your progress in meeting one of these quality gates.

7. Identify Assumptions Used in Preparing the Project Plan

A typical systems or software development program plan includes a number of assumptions about progress, quality, and resources. Assumptions made during program planning are excellent informational needs for the measurement process. If the assumption is not realized, then many of the resulting schedule and resource plans may need to be examined, or re-planned.

For example, when performing an estimation using the Cost Constructive Model (COCOMO), the estimated number of lines of code (or other software sizing measure such as function points) is a primary driver in establishing the amount of effort. If your project exceeds the number of lines of code during development, this may indicate that more effort is needed during the *down stream* software activities such as unit testing, system testing, or integration.

8. Identify Resources Consumed and Products Produced to Understand Process Performance

At the beginning of initiating a measurement process, your organization will typically not have historical process data. In such cases, one of your goals should be to understand the behavior of the processes in the systems or software life cycle. Consider that each process in the life cycle consumes resources and produces a product (either an internal or a customer product). You should establish basic measurements to determine how many resources are being consumed and how much of the product is being produced. You might consider doing this for the processes that consume the majority of your budget or schedule.

9. Identify the Information Needed to Satisfy Organizational Policy

In many system or software shops, managers are required to use specific techniques in monitoring and controlling their programs. In large defense programs, for example, an earned value management system is required. When managers are required to use specific management techniques, the organization should provide the data that managers need to effectively apply the technique. The measurement process is the method that the organization uses to deliver the information that managers need to use the technique.

For example, many defense organizations must use an earned value management system. In support of this, the measurement process should deliver required cost and schedule status information to the program in the form of cost performance index, schedule performance index, to-complete performance index, earned value (and its components), and variance at completion. Without a measurement process to ensure that the earned value data is collected, analyzed, and delivered in a timely fashion, even a very useful technique such as earned value can be inconsistently or (often) incorrectly applied.

Program or site policy and standards documentation provides many informational needs for the measurement process. While setting up a measurement process, analyze the systems and software management standards and policy to see what management techniques have been, or are being, mandated. Then, extract the informational needs from these mandated standards and address them during measurement process implementation.

Summary

Identifying informational needs is the first step in establishing an effective measurement process. The techniques explained above provide tools for extracting the real informational needs within your organization. Once all informational needs are identified, you can assign a relative priority to them, in case you need to balance the informational needs and the resources available to the measurement process. The measurement process will refine those informational needs into appropriate measurement activities, specific measures, and information products. Over time, you should review the effectiveness of the information products and individual informational needs as your organization adopts new technology and processes.

This approach for identifying informational needs ensures that the measurement information you and other managers receive is effective in helping you monitor and control your programs. By focusing measurement on true informational needs, managers are better armed to monitor and control their programs and to assess the likelihood of an on-time and on-budget completion. In addition, by saving managers time in gathering and analyzing the information they need to manage, managers can spend more time on their real role: decision making.◆

References

1. Department of Defense and U.S. Army. Practical Software and Systems Measurement Guidebook. Version 4.0b. Oct. 2000.
2. Software Engineering Institute. Capability Maturity Model® Integration for Systems Engineering /Software Engineering/Integrated Product and Process Development/Acquisition. Version 1.02d. Carnegie Mellon University. Dec. 2000.
3. Caputo, Kim. CMM® Implementation Guide. Addison-Wesley, 1998.
4. DeMarco, Tom. Controlling Software Projects. Prentice Hall, 1982.
5. Zachary, Pascal. Showstopper! The Free Press/Macmillan, 1994: 243-255.

About the Author

Peter Baxter is the development manager at Distributive Software, where he directs measurement services, products, and training. For the past eight years, he has assisted numerous government and commercial organizations in planning and implementing measurement programs. He is a frequent speaker and trainer on the subject of applying measurement to software, information technology, and systems engineering. He is the current chair of the International Council of Systems Engineering. Measurement Working Group and a member of ISO Subcommittee on Systems and Software Engineering . The author welcomes comments and discussion on this article.

Distributive Software
2300 Fall Hill Avenue, Suite 100
Fredericksburg, Virginia 22401
Phone: (540) 372-4500
Fax: (540) 372-6497
E-mail: pbaxter@distributive.com

Coming Events

July 18-20
Shareware Industry Conference
St. Louis, MO
www.sic.org

July 22-25
Joint Advanced Weapons Systems Sensors, Simulation, and Support Symposium (JAWS S3)
Colorado Springs, CO
www.jawswg.hill.af.mil

July 22-26
6th Annual Practical Software Measurement Users' Group Conference
Keystone, CO
www.psmsc.com

August 19-22
The 2nd Software Product Line Conference
San Diego, CA
www.sei.cmu.edu/SPLC2/

September 9-13
International Conference on Practical Software Quality Techniques (PSQT) 2002 North and International Conference on Practical Software Testing Techniques (PSTT) 2002 North
St. Paul, MN
www.psqtconference.com

November 18-21
International Conference on Software Process Improvement
Washington, DC
www.software-process-institute.com

April 28-May 1, 2003
Software Technology Conference 2003
STC
Salt Lake City, UT
www.stc-online.org

JAD on a Shoestring Budget

Dr. Mario J. Spina
The George Washington University

John A. Rolando
Science Applications International Corporation

Why is it seemingly so difficult to adequately address that first and all-important development phase – gathering and defining software requirements? The use of Joint Application Development offers more than a worthwhile and proven approach; it can be adapted to accommodate the business and administrative challenges in requirements gathering when there is seldom enough time and resources available to do it right the first time. The following article discusses the details of how the Science Application International Corporation in McLean, Va. was able to inexpensively bring together users and developers to define complex and disparate requirements in a disciplined and effective manner. This laid a foundation for successful integration of application needs with existing commercial off-the-shelf and government off-the-shelf software tools and products.

The complexities of software development provide fertile ground for debate regarding which activities constitute its most critical steps and processes. However, many discussions on this subject suggest that the final frontier for successful software development is now, and may continue to be, the requirements gathering process.

For better or worse, gathering requirements demands involving the software application users, many of whom are neither educated nor experienced in the software development enterprise. Many software customers mistakenly believe that the up-front time spent in requirements gathering and analysis simply translates to an equivalent time delay in product delivery. This belief holds irrespective of studies and evidence showing that costs associated with correcting errors traceable back to poor up-front requirements, after fielding, can range from 68 to 200 times higher than the preventive costs associated with catching the errors during the requirements analysis phase [1].

With so much at stake, it is still surprising to learn how divergent the methodologies presented in textbooks and software journals are when discussing ways to elicit and rationalize software requirements. Few software engineering texts seem to provide much detail on how to elicit requirements, or otherwise may do so in one or more brief chapters or paragraphs offering only a short list of steps to consider in the process. Although referred to by a variety of names, a software requirements elicitation process called Joint Application Development (JAD) recognizes that requirements gathering is a very social endeavor [2]. It proposes a disciplined process for collecting, understanding, and organizing the innumerable and enigmatic user perspectives inherent in a clear formulation and conceptualization of software needs.

> *“Costs associated with correcting errors traceable back to poor up-front requirements, after fielding, can range from 68 to 200 times higher than the preventative costs …”*

Traditional systems design can result in disparate and conflicting requirements subject to mixed interpretations, and derived from the frequently limited participation of subject matter experts (who might also be the users). This precedes lengthy attempts to get feedback needed to reconcile and clarify the inevitable inconsistencies. As this article will attempt to show, the JAD approach is a proven course of action that may be ideally suited toward generating and organizing more clear and complete requirements from multiple users, when compared with traditional means of extraction [3].

Short History and Characteristics

JAD’s lineage traces back to business systems’ planning methodology developed by IBM in the mid-1970s. In addition to defining software requirements, this methodology also has been successfully applied to business planning, manufacturing, strategic planning, cost estimating, test planning, and other domains. While variations to its use have evolved, some developers and software experts still consider it to be the best method for collecting requirements from the perspective of the users, customers, or customer advocates – classifying JAD as best practice for software projects [4].

In brief, JAD is essentially a structured workshop approach calling for a detailed agenda, a facilitator, comprehensive visual aids, and detailed record keeping. The characteristics of JAD, and keys to its success, are having committed participants, group cohesion, and the organization and structured setting just described.

Implementing a JAD session involves five phases. First, the project must be clearly defined in terms of purpose, functions, the team participants, and schedule. Second, some background research must be completed on user requirements and any anticipated problems and unique processes that might be required. Third, and particularly important, comprehensive visual aids should be utilized to help all participants better understand and visualize needs. Fourth, the session itself must be guided, all issues resolved, and all agreements well documented. Fifth, the session should be written up in a formal after-action report to be the basis for the software requirements [5].

Significant emphasis must be placed on two factors in this process. First, time, thought, and resources should be invested in having the best visualization tools as possible for the workshop [6]. Second, the commitment of top management is critical, especially in terms of the quality of, and direction given to, the session participants [7].

The Wireless Mobile Worker

Science Applications International Corporation (SAIC) recently implemented the JAD concept in a unique way. The project is a valuable industry case study because it was successfully performed with very little budget. As a systems integrator, SAIC is not traditionally in the business of wholly developing commercial software products. This particular project focused upon the wireless mobile-worker market for online, automated procedures to perform test and inspection in the nuclear power industry.

Wireless mobile-worker technology in 1999 had an almost negligible installed base; capturing the customers' needs was critical to its acceptance in the market. To accurately do this, SAIC had to solve three problems. First, it had to have a rigorous forum from which to explain the concept and query its customers. Second, it had to get the customers to dialogue collectively with their engineers. Third, the SAIC engineering team had to find the funds required to support a conference for a thorough customer inquiry.

All of these problems were answered by one simple business development concept: teaming with industry. SAIC identified the key technologies and vendors providing solutions for elements of the procedure's automation business process. The areas of critical concern for SAIC's nuclear power plant customers were online interactive procedure development, document configuration management, real-time mobile computing, and data recording and reporting. No single vendor or product was able to offer the full life cycle solution.

SAIC reasoned that integrating existing products rather than attempting to re-invent the wheel could drastically reduce the costs of a full-scale software product development. Unfortunately, bringing the various vendors and products together in a full-scale integration was considered unattainable without some guaranteed number of paying customers. How better to guarantee customer involvement than to include them in the design and get up-front commitments? To help shepherd both the product vendors and power plant customers together, SAIC evolved the concept of a JAD-based conference called the Procedure Automation Consortium.

For the conference, vendors were to supply their products and the necessary application program interface code, and SAIC was to perform the role of technology solutions architect and systems integrator. To communicate the idea and kick-off the conference, SAIC hosted the JAD meeting in cooperation with a nationally recognized nuclear utility services provider. The three-day event, held in May of 1999 at the upscale Arlington Hilton Hotel, brought 12 product vendors together with 16 power plants for a total participation of approximately 80 industry professionals, split evenly between customers and vendors.

JAD in Action

Each day of the JAD conference began with a keynote talk by a prominent industry figure having a strong grasp of technology trends in the field. The first two mornings, attendees viewed product presentations from each vendor in four separate conference rooms, one for each of the business process specialties: procedure development, document management, mobile procedure execution, and data recording and reporting. The first two afternoons attendees were asked to indicate what they liked and did not like about the products they had seen during the morning sessions.

Professional facilitators worked to cultivate customer and vendor interchange and turn customer comments into system requirements. The facilitators were supported by stenographers and employed an on-screen requirements database. The full-screen view of each requirement, displayed as simple, stand-alone *shall* statements allowed customer groups an opportunity to specify, negotiate, and validate the precise phrasing of each system requirement. The facilitators focused upon building requirements consensus through technical discussion and point negotiations.

> *"Professional facilitators worked to cultivate customer and vendor interchange and turn customer comments into system requirements."*

On the third and final day, the results of the previous two days were consolidated and presented in summation to the collective audience. Before lunch, customers were asked to rate the products they had seen, identify a value and a desire to purchase such products, and indicate their willingness to fund a political action committee effort to develop a product that would achieve the set of requirements identified at the conference. As a structured workshop approach, JAD is normally performed in relatively small classroom-sized settings. Having 80 participants placed a premium on visual aids, experienced facilitation, and opinion management. The large full-screen display was critical.

Consortium Results

The full affair was well done with three quality meals each day and full-time coffee, tea, and soda service. The total bill for the event was approximately $35,000, but SAIC paid only a small portion of the costs. The product vendors were willing to sponsor meals and advertisement banners, and attendees were willing to pay a small $100 conference fee. Overall, the venture was well worth the effort, not only for its informational value but also for the industry goodwill it engendered and the networking opportunities it offered.

The data from the survey was extensive. Customers had not only indicated their requirements and the products they preferred, but also indicated an initial willingness to purchase the associated software solution. SAIC was thereby armed with the information needed to select integration partners, the product requirements, the purchase price, and the business probability data from which to make a project go/no-go decision.

With this information, a COTS application architecture was defined; key vendors provided cost estimates for the licensing, interface development, and integration of their products into a final solution. With an assumption that initial development and roll-out costs could be spread across the set of customers who indicated a high likelihood of product purchase, a per customer solution cost was estimated. From this estimate, the SAIC architect was quickly able to determine that, unfortunately, product costs significantly exceeded customer perceived value. Accordingly, this product development effort was cancelled.

While this JAD did not result in a successful software product development, it presented some valuable information and experience. The effort showed how the JAD approach could be used when there is a need to develop and refine a software solution that integrates an existing product base. It also showed how a software requirements elicitation involving software users, product vendors, and a systems integrator could be accomplished on a shoestring budget. Lastly, there was the advantage of having multiple potential customers together to analyze and assess the demand and economic viability of further development.

Epilogue

JAD is, of course, one of a multitude of requirements elicitation techniques. We have found no actual software developmental data comparing the use of JAD with alternative procedures and techniques in terms of relative successes or failures. In addition, since all projects are somewhat unique, we believe any comparative empirical data would somehow have to be adjusted for the multitude of other diverse variables to be meaningful.

It would seem intuitive that there would be advantages in having a dedicated gathering of users, developers, and customers together in a structured setting, compared with shorter piecemeal sessions or multiple one-on-one sessions. However, it is also easy to imagine circumstances whereby just the converse would be true, i.e., that shorter piecemeal sessions or multiple one-on-one sessions would be better if, for example, the JAD facilitator was somehow skewing inputs directly or indirectly via the recording process [8]. Thus, before any reader commits to using the JAD approach, we suggest they perform their own analysis and thought. We offer the following list of sources and Internet sites for further research.◆

References

1. Weigers, Karl E. Software Requirements. Redmond, Wa.: Microsoft Press, 1999: 15.
2. Wood, Jane, and Denise Silver. Joint Application Development. New York: John Wiley & Sons, Inc. 2nd ed., 1995: 30-31. Here are some synonymous examples: facilitated meeting, facilitated session, facilitated design, joint application design, joint application development, joint application review, joint application planning, et al.
3. Ibid. 5-6.
4. Cline, Allan. "Joint Application Development (JAD) for Requirements Collection and Management." White Paper. Carolla Development. Available at <www.carolla.com/wpjad.htm >.
5. Weigers, Karl E. Software Requirements. Redmond, Wa.: Microsoft Press, 1999: 8-9.
6. August, Judy H. Joint Application Design: The Group Session Approach to System Design. New Jersey: Yourdon Press, 1991: 5.
7. Wood, Jane, and Denise Silver. Joint Application Development. New York: John Wiley & Sons, Inc. 2nd ed., 1995: 166.
8. Christel, Michael G., and Kyo C. Kang. "Issues in Requirements Elicitation." Technical Report CMU/SEI-92-TR-12. Sept. 1992, ESC-TR-92-012. Available at <www.sei.cmu.edu/publications/documents/92.reports/92.tr.012.html>.

Additional Reading

1. August, Judy. Joint Application Design. Englewood Cliffs, N.J.: Yourdon Press, 1991.
2. Garner, Rochelle. "Why JAD Goes Bad." Computerworld 10 April 1995.
3. Hollander, Nathan, and Naomi Mirlocca. "Facilitated Workshops: Empowering the User to Develop Quality Systems Faster." Industrial Engineering Oct. 1993.
4. Knowles, Anne. "Peace Talks: Joint Application Development." PC Week 11 Dec. 1995.
5. Leventhal, Naomi. "Using Groupware Tools to Automate Joint Application Development." Journal of System Management Sept.-Oct. 1995.
6. Martin, James. Rapid Application Development. New York: MacMillan Publishing Company, 1991.
7. McConnell, Steve. Rapid Development. Redmond WA: Microsoft Press, 1996.
8. Wetherbe, James C. "Executive Information Requirements: Getting It Right." MIS Quarterly Mar. 1991.
9. Wood, Jane and Denise Silver. Joint Application Development. New York: John Wiley & Sons, Inc. 2nd ed., 1995.

Web Sites

- <www.utexas.edu/hr/is/pubs/jad.html>.
- <www.dci.com/events/jad>.
- <www.credata.com/research/jad.html>.
- <www.russellmartin.com/courses/JAD.htm>.
- <www.carolla.com/wp-jad.htm>.
- <csweb.cs.bgsu.edu/maner/domains/RAD.htm>.
- <www.trainersdirect.com/outlines/JAD.htm>.
- <www.sisg.com/JAD.htm>.
- <www.verhoef.com/cs/njrad.htm>.
- <www.computer.muni.cz/cspress/CATALOG/rs00072.htm>.
- <www.mgrossmanlaw.com/articles/mhtl/joint_application_development.htm>.

About the Authors

Mario J. Spina, D.Sc., is a part-time faculty for The George Washington University School of Business and Public Administration and the School of Engineering and Applied Science, as well as deputy division manager for Science Applications International Corporation's (SAIC) Aerospace Technology Applications Division. As a new product development manager for SAIC, Dr. Spina led the company's charge into the wireless mobile worker software market. He has a doctorate of science degree in engineering management from The George Washington University, a master's of science degree in electrical engineering from the University of Southern California, and a bachelor's of science degree in mechanical engineering from California State University, Northridge.

310 Cabin Rd. SE
Vienna, VA 22180
Phone/Fax: (703) 242-6732
E-mail: mspina@gwu.edu

John A. Rolando is currently a senior operations research analyst for Science Applications International Corporation (SAIC) Information Technology Solutions Group. He has been directly supporting communications interoperability analyses and the development of two detailed communications models for the Command, Control, Communications, and Computer Systems Directorate, Joint Staff, the Pentagon, since 1996. He has a bachelor's degree in economics from LeMoyne College, Syracuse, a master's degree in public administration from the University of Northern Colorado, and a master's degree in business administration from Marymount University, Arlington, Va.

Science Applications International Corporation
1710 SAIC Drive
McLean, VA 22102
Phone: (703) 676-4682
Fax: (703) 356-2534
E-mail: john.a.rolando@saic.com

National Information Assurance Acquisition Policy

This article reviews the national policy governing the acquisition of information assurance (IA) and IA-enabled information technology products and becomes effective July 1, 2002. The National Security Telecommunications and Information Systems Security Policy No. 11 was issued by the National Security Telecommunications and Information Systems Security Committee in January 2002.

The National Security Telecommunications and Information Systems Security Policy No. 11 (NSTIS-SP No. 11) was written to address the problems associated with acquiring commercial off-the-shelf (COTS) security and security-enabled information assurance (IA) products. While this policy includes helpful ideas and information to successfully complete the acquisition process, it is not a stand-alone document. Its origin can be tied back to Department of Defense (DoD) Information 5000.2-R. The current agency implementing this policy is Global Information Grid 6108510. There are also emerging policies awaiting DoD approval: directive 8500.aa and instruction 8500.bb.

Accordingly, the NSTIS-SP No. 11 has been developed as a means of addressing these problems for those products acquired for national security applications. The policy also rightfully points out that protection of systems encompasses more than just acquiring the right product. Once acquired, these products must be integrated properly and subject to an accreditation process, which will ensure total integrity of the information and systems to be protected.

The Policy

IA shall be considered as a requirement for all systems used to enter, process, store, display, or transmit national security information. IA shall be achieved through the acquisition and appropriate implementation of evaluated and validated government-off-the-shelf (GOTS) or COTS IA and IA-enabled information technology (IT) products. These products should provide for the availability of the systems; ensure the integrity and confidentiality of information, and the authentication and non-repudiation of parties in electronic transactions.

Effective Jan. 1, 2001, preference shall be given to the acquisition of COTS IA and IA-enabled IT products (to be used on systems entering, processing, storing, displaying, or transmitting national security information) that have been evaluated and validated, as appropriate, in accordance with the following:

- The International Common Criteria for Information Security Technology Evaluation Mutual Recognition Arrangement.
- The National Security Agency (NSA)/National Institute of Standards and Technology (NIST) National Information Assurance Partnership (NIAP) Evaluation and Validation Program.
- The NIST Federal Information Processing Standard (FIPS) validation program.

The evaluation/validation of COTS IA and IA-enabled IT products will be conducted by accredited commercial laboratories, or the NIST.

By July 1, 2002, the acquisition of all COTS IA and IA-enabled IT products to be used on the systems specified shall be limited only to those that have been evaluated and validated in accordance with the criteria, schemes, or programs specified in the three sub-bullets.

The acquisition of all GOTS IA and IA-enabled products to be used on systems entering, processing, storing, displaying, or transmitting national security information shall be limited to products that have been evaluated by the NSA, or in accordance with NSA-approved processes.

Normally, a complementary combination of IA/IA-enabled products is needed to provide a complete security solution to a given environment. Thus, in addition to employing validated IA/IA-enabled products, a solution security analysis should be conducted as part of the certification and accreditation process. In support of this, NSA shall provide guidance regarding the appropriate combinations and implementation of GOTS and COTS IA and IA-enabled products.

Subject to policy and guidance for non-national security systems, departments and agencies may wish to consider the acquisition and appropriate implementation of validated COTS IA and IA-enabled IT products. The use of these products may be appropriate for systems that process, store, display, or transmit information that, although not classified, may be critical or essential to the conduct of organizational missions, or for information or systems that may be associated with the operation and/or maintenance of critical infrastructures as defined in Presidential Decision Directive No. 63 (PDD-63), Critical Infrastructure Protection.

Responsibilities

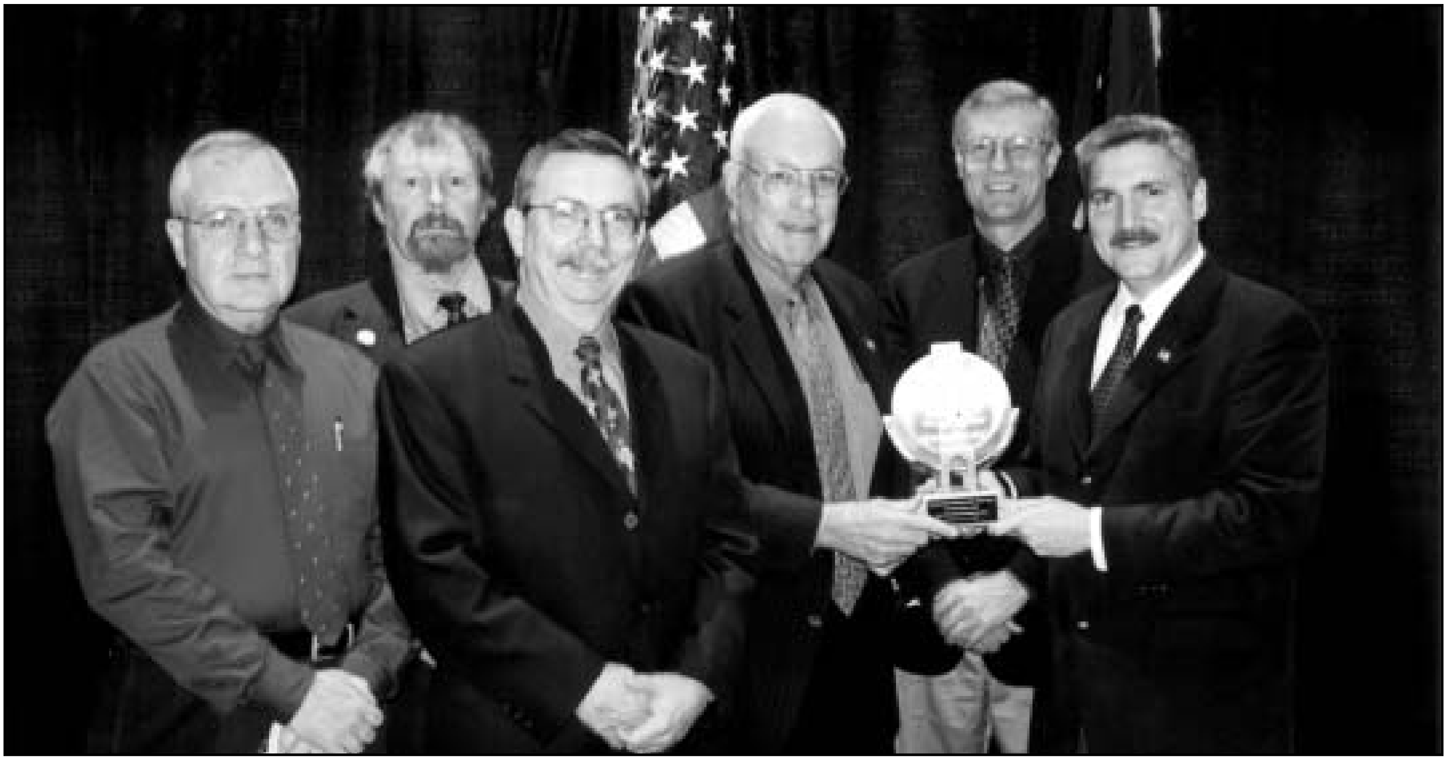
Heads of U.S. departments and agencies are responsible for ensuring compliance with the requirements of this policy.

Exemptions and Waivers

COTS or GOTS IA and IA-enabled IT products acquired prior to the effective dates prescribed herein shall be exempt from the requirements of this policy. Information systems in which those products are integrated should be operated with care and discretion and evaluated/validated IA products and solutions considered as replacement upgrades at the earliest opportunity.

Waivers to this policy may be granted by the National Security Telecommunications and Information Systems Security Committee (NSTISSC) on a case-by-case basis. Requests for waivers, including a justification and explanatory details, shall be forwarded through the director, National Security Agency (DIRNSA), ATTN: V1, who shall provide appropriate recommendations for NSTISSC consideration. Where time and circumstances may not allow for the full review and approval of the NSTISSC membership, the chairman of the NSTISSC is authorized to approve waivers to this policy, which may be necessary to support U.S. government operations that are time-sensitive, or where U.S. lives may be at risk.

For additional information or clarification, contact the National Security Agency at (410) 854-6805, or toll free at 1 (888) NSTISSC, or e-mail <nstissc@radium. ncsc.mil>.◆



Force XXI Battle Command Brigade developed by TRW Tactical Systems Division. The award was accepted by (from left to right) Tom Carter, Hal Hart, John Dowdee, Lt. Gen. (Ret.) Otto Guenther, and Clark Lewis.

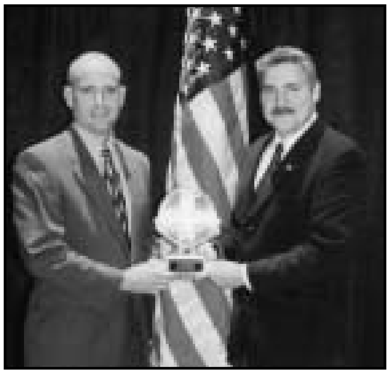
Higher Authority Communications/Rapid Message Processing Element developed by Detachment 1, Ogden ALC. The award was accepted by SrA Joshua Babcock (left) and Captain David Selnick (right).

CrossTalk Presents Top 5 Awards at Software Technology Conference

Standard Terminal Automation Replacement System developed by ATB-230 STARS Product Team. The award was accepted by Alan Feinberg.

The winners of CrossTalk's 2001 Top 5 U.S. Government Quality Software Projects were presented with their awards at the 2002 Software Technology Conference (STC) held recently in Salt Lake City. Lt. Col. Glenn A. Palmer, program director for Computer Resources Support Improvement Program at Hill Air Force Base, introduced the representatives from each project to more than 2,200 STC conference attendees at a morning co-sponsors' panel discussion. Joe Jarzombek, deputy director of Software

CONTINUED ON PAGE 31



System Configuration Set 15C developed by F/A-18 Advanced Weapons Lab. The award was accepted by Keith Heflin.

Data Capture System 2000 developed by Lockheed Martin. The award was accepted by Sean Murphy.

Information Insurance

Yeah, I know. The topic of the issue this month is Information Assurance. So either I can't spell very well, or I don't know the difference between *insurance* and *assurance*. Actually, both words pretty much mean the same. In fact, I remember a door-to-door salesman that was trying to sell me a policy years ago. He said, "With our insurance, you have the assurance that your loved ones will be well taken care of."

Remember the days of door-to-door salesmen (and saleswomen)? Funny how you reminisce about strange things. My mother always felt sorry for them and wanted to invite them in for something to drink and a piece of cake. I can hear her now, "That poor man, trying to make a living in this heat." My dad, on the other hand, was more of a "thanks but no thanks" then shut-the-door-in-their-face type of person.

You understand, of course, that those same people who used to come to your door now work as mass marketers for some company on the Internet. I mean, after all, why spend your time going to one house at a time, when you can instead, in just a few short keystrokes, affect the lives of 20 million people with "spam" mail.

Spamming e-mail is, unfortunately, habit-forming. You used to get the e-mail spam only from salespeople interested in refinancing your house, getting you to change Internet companies, or something similar. Now, every business seems to think that sending out e-mail to everybody they know will somehow make their company successful.

I've traveled a lot and can usually count on five to 10 spam messages per day while on the road. Unfortunately, I also am stuck with slow hotel-speed modem connections, so it's pretty well guaranteed that anybody who sends me a three megabyte file of cute graphics and color is going to clog my mail program. Of course, this usually happens when I am in a hurry and need to download something important.

HEY, COMPANIES THAT ADVERTISE BY SENDING OUT MASS E-MAIL, DO YOU WANT ME TO BUY FROM YOU?? Then send me an e-mail saying, "Company XXX will never advertise on e-mail to you, even though we have your address." Send it as a text file, about 1K in length. Trust me – if you send a large file with lots of graphics, I'm going to tell everybody I know that whatever product you sell sucks swamp water.

Sorry. I got off the subject for a minute, but I feel "much better now." The title of my column is called "Information Insurance." I am about to create a column that system administrators will be posting worldwide, because I'm about to warn you who is responsible for insuring that your data is "safe." You are.

I had the misfortune of having two separate laptops crash on me within the last three months. In both cases, there was a hardware failure and the hard drive "went to the data graveyard in the sky."

In both cases, our local system administrators were able to set me up with a new machine and new operating system within a day (thanks, again, Randy and Geoff). In both cases, all of my data on the old machine was lost. Was I really upset? Not too much. Why? Because I usually burn a weekly CD to back up all of my data. Total work lost each time? About three days of file updates.

To quote an old television commercial, "It's 10 o'clock. Do you know where YOUR [data] are?" Let me be honest; do you think that your system administrators really have nothing better to do than perform instantaneous backups of your data? You know, the system administrators are busy just keeping the network running and installing the patches, updates, new drivers, and other essential software. If they do have time to perform backups, do you know how often? And what if your machine dies just before the next scheduled backup?

To be safe, you need to perform regular backups on your own. You don't need to save everything, of course – too much room. In fact, it used to be that all of my "critical" data could fit on a few floppies. Now, my cartoon collection (which is absolutely critical for my PowerPoint slide presentations) takes up about 100 MB. Luckily, the cartoons don't change much, so once a CD is burned, that backup is good for months. If you don't perform personal regular backups of your critical data, then what critical files are you going to be missing when a recovery is done?

In short, this column should serve to remind you that information assurance is not just something at a global level that affects large-scale software. Information insurance is sort of like personal information assurance. Just like real insurance, it helps you rest well at night, secure in knowing that you have protected those (files) dear to you.

By the way – you are lucky that we have had too many Backtalk columns in recent months that were takeoffs on songs. The old Baptist hymn "Blessed Assurance" kept running through my head, and I really think I could work up a great set of data-based lyrics. Whoops; the editor says I'm out of space – too bad.

– **David A. Cook**
david.cook@hill.af.mil
Software Technology Support Center

Continued From Page 30

Intensive Systems Office of the Secretary of Defense/AT&L, the department sponsoring the contest, presented the awards.

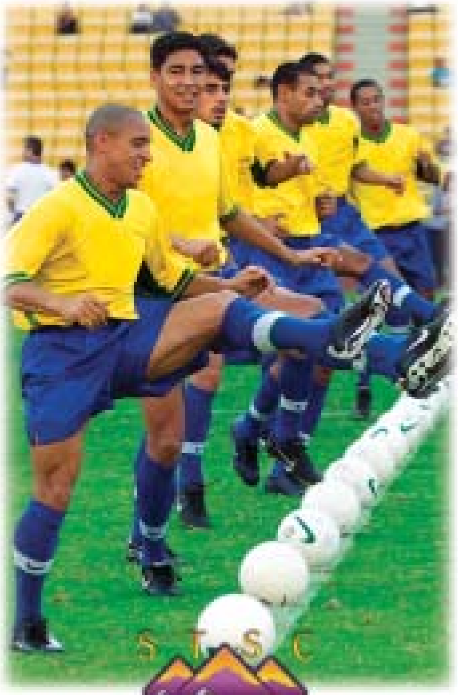
The intent of this search was to recognize outstanding performance of software teams and to promote best practices. These Top 5 project winners were selected from 87 nominations in this first government-sponsored event. Each nomination was preliminarily scored based on customer value, performance, and technical value. The customers of the highest scoring projects were then contacted to ensure their satisfaction with the nominated projects. Using this information, the top 16 projects were chosen as finalists and sent to a board of judges who selected the top five software projects.

Complete articles on each winning project were published in the January 2002 issue of CrossTalk, *The Journal of Defense Software Engineering*. This, and other back issues, can be found on the CrossTalk Web site at <www.stsc.hill.af.mil/ crosstalk>.

CrossTalk is now accepting nominations for the 2002 Top 5 contest. Applications are available at the Web site listed above or at <www.stsc.hill.af.mil>.◆

PROCESS IMPROVEMENT

getting YOUR team in line



The *Software Technology Support Center* (STSC) helps you improve your processes by identifying root causes of your problems; constructing simple, practical solutions; and creating ownership of the solution. In other words, we help determine what to change, what to change to, and how to cause the change. But how do you get started?

Call us. The STSC can help plan process improvement at any level. Once you decide what you need, we are the group to help implement the solution, whether it's implementing the Capability Maturity Model®, setting up a Management Steering Group, or designing a Software Engineering Process Group infrastructure. We help you understand how to implement process improvement.

Don't start your process improvement without a STSC mentor.

We have been in the trenches and have the scars to prove it. Call us first. Whether your organization is big or small, just starting a project or embattled in difficulties, we can help. We bring hands-on experience.

OO-ALC/TISE • 7278 4th Street • Hill AFB, UT 84056 • 801 775 5555 • FAX 801 777 8069 • www.stsc.hill.af.mil

Sponsored by the
Computer Resources
Support Improvement
Program (CRSIP)

Published by the
Software Technology
Support Center (STSC)

CrossTalk / TISE
7278 4th Street
Bldg. 100
Hill AFB, UT 84056-5205