The Internet: Brainchild of the DoD

Forrest Brown
Managing Editor

A few years back the Department of Defense (DoD) allegedly paid over $400 for a hammer. The cost of a toilet seat for the space shuttle was said to be over $1,000. Just before the Gulf War in 1991, a group of U.S. congressmen claimed we had "gold plated" our military by spending millions of taxpayer dollars unwisely. Soon after that, many watchdog groups began to keep a closer eye on how the DoD spent its dollars.

Until recently, I, too, was concerned that maybe the taxpayer was not getting a fair deal, since I am also a taxpayer. Then, while I was surfing the Web one day, I ran across an article on the history of the Internet. I was interested to learn that over 30 years ago, the RAND Corporation was commissioned by the DoD to come up with a way for the U.S. government to communicate after a nuclear war attack. The article states that "any central authority, any network central citadel, would be an obvious and immediate target for an enemy missile" and "the center of the network would be the very first place to go." Consequently, an Internet was developed to "have no central authority"; "all the nodes in the network would be equal in status to all other nodes, each node with its own authority to originate, pass, and receive messages" (http://w3.aces.uiuc.edu/AIM/scale/nethistory.html). This, in essence, was the birth of what is now one of the fastest-growing technologies in the world.

Today, much of our daily work involves the Internet. Statistics from the Graphics, Visualization & Usability Center's eighth World Wide Web user survey (http://www.gvu.gatech.edu/user_surveys/survey-1997-10) indicate that 85 percent of the respondents use the Web daily; the largest number of respondents use it one to four times daily (45 percent), whereas 41 percent use it more frequently. Also, daily Web access by expert users is considerably higher (94 percent) than that of novice users (78 percent). Of 2,921 people responding, 38 percent spend up to five hours a week using their browsers, while 35 percent spend six to 10 hours a week on line (http://www.ocean.ic.net/ftp/doc/nethist.html). The data also strongly indicate that Internet use is continuing to rise drastically.

Where does the Internet go from here? Recently, in *Federal Computer Week*, Heather Harreld reported that the Next Generation Internet Program, which is currently being discussed before the U.S. Senate, "has a goal of increasing the present speed of the Internet by as much as 1,000 times." Just imagine what we can do then! Greg Meyer's article on one of the new Internet technologies, XML (page 6), illustrates that this new format soon will support intelligent information management on the Internet, improving on the current HyperText Transfer Markup Language and Standard Generalized Markup Language formats. On the horizon for Intranets (implementation of Internet technologies within an organization) is Java database connectivity, interactive forms, and collaborative design and reviews with off-site customers (see "Developing an Intranet for a Small Unit," p. 3).

In light of these developments, we can conclude that good and useful things have come from our defense and military efforts (*CROSSTALK* is one example) and that we can undoubtedly continue to receive benefits beyond national security from our military organization. ◆

Letter to the Editor

GSAM and *CROSSTALK* Provide Needed Resources

Your *Guidelines for Successful Acquisition and Management of Software-Intensive Systems (GSAM), Version 2.0, June 1996* has been immensely beneficial in carrying out various assignments. It is a rich encyclopedic resource, which is supplemented by your monthly *CROSSTALK*. Together they form a valuable single source of reference for almost everything one needs on various phases of systems development lifecycle.

Adepu Bikshapathi
EER Systems, Inc.

Readers can comment on articles printed in CROSSTALK. Please limit your comments to under 250 words. Include your name, address, and telephone number. We will withhold your name if requested. Your letter may be edited for grammar, content, and space. Please send correspondence to CROSSTALK, OO-ALC/TISE, 7278 Fourth Street, Hill AFB, UT 84056-5205 or E-mail to senior_editor@stsc1.hill.af.mil.

Developing an Intranet for a Small Unit

Capt. Dan Stormont
U.S. Air Force

The Theater Air Command and Control Simulation Facility (TACCSF) is a fairly small unit that works closely with contractors to develop simulation software and scenarios. The small size of the unit and the interaction with contractors prompted us to develop efficient ways to share data and documentation. This article discusses the approaches the TACCSF has taken to set up an Intranet using limited resources. It addresses the use of shared memory devices and the problems associated with that approach as well as the efforts undertaken to develop a browser-based (Intranet) paradigm. The lessons learned from these attempts are presented, as are our current efforts and future plans to develop the most cost-effective solution that will satisfy our needs.

How do you share information in your office? Pin pages on a bulletin board? Send E-mail out to everyone you think would be interested? Attach a routing slip to something and drop it in someone's in-box, never to be seen again? Write important meetings on a centrally located calendar? Although there is a need at times for formal routing of correspondence and face-to-face coordination of meetings, much of the information passed around in the typical office would best be disseminated by having it readily available on everyone's desktop computer. Recognizing these issues at the TACCSF motivated us to develop an efficient, usable Intranet for our organization. This article discusses the trials, tribulations, and lessons learned from these efforts.

About TACCSF

The TACCSF is an Air Combat Command-operated advanced distributed simulation facility. We are physically separated from our parent units: the 505th Command and Control Evaluation Group (CCEG) and the Air and Space Command and Control Agency.

The TACCSF team is uniquely composed of military members of Detachment 4, 505th CCEG, development contractors working for Lockheed Martin Corporation, and support contractors working for Scientific Research Corporation. The team works in the same building and must work closely on development projects for external customers interested in using our simulation facility. We have two PC networks—one operated by Detachment 4 and one operated by Lockheed Martin—that are used to share project information but must limit access to competition-sensitive or government-only information.

Another constraint is that Intranet site development and maintenance is only an additional duty for the people working on it, and funding for equipment and software is usually based on the amount of customer funds we receive in a year. We have a small unit with part-time Intranet employees operating with limited funds. Does this sound like your organization?

Why an Intranet?

Why did we want to develop an Intranet? We wanted to make as much information as possible available to people in the organization: information such as conference room schedules, important dates for the organization (Commander's Calls, etc.), and project information (schedules, funding charts, test plans, etc.). Most people are familiar and comfortable with using a Web browser, so an Intranet seemed to be a natural medium to share information.

Implementation Goals

What should an Intranet offer to be useful to a small unit like TACCSF? Most important, it needs to be easy to use. "Ease of use" is a phrase that tends to be ill-defined and overused; I define it as the ability to find the desired information (if it is available) with a minimal number of page changes. To make an Intranet easy to use requires a great deal of design and planning to logically organize and present the available data.

The Intranet also needs to limit access to sensitive information and be accessible to everyone who has a right to the information contained in it. This requires the use of password protection and possibly secure HyperText Markup Language (HTML) to protect the sensitive information.

Finally, the Intranet needs to be easy to maintain. The best Intranet site in the world is useless if it takes too long to update or demands that all changes go through a single choke point.

Using a Shared Memory Approach

When we first tried to share information at TACCSF, we used what I call a "shared memory" approach. This meant that people who wanted to share information, i.e., a document or a briefing, would place it in a shared memory area on the local area network (LAN) server and advertise its availability to people who might want to view the data.

There are a number of shortcomings with this approach. The biggest challenge can be to find the desired data. It does not take long for directories to proliferate in the shared area, which makes files extremely difficult to find unless you know the exact file path. Also, unless you have a file plan for your shared memory, you start to get several variations in organization and labeling of directories and files.

The shared memory approach is also a burden on the LAN administrator who must determine appropriate permissions on a directory-by-directory or often a file-by-file basis. The LAN administrator also has to juggle server

memory management since some users post more shared files than others.

First Attempt

Difficulties with the shared memory approach led to our first Intranet attempt—the TACCSF Intranet Software Engineering Environment (ISEE). The intent of ISEE was to share information concerning TACCSF projects. The site was organized by project and would contain information such as on-line documentation, project management data (schedules, financial reports), and briefings.

The biggest challenge with ISEE was the lack of good Web development tools at the time. Most of the HTML pages for ISEE were developed by hand using HTML editors that require the author to work directly with the HTML source code and refer to an HTML reference guide. This made page development tedious and time-consuming.

There were, however, some good ideas that came from the ISEE experience. We discovered that the best way to minimize page changes was to nest lists of data from the most general to the most specific. (This may seem obvious, but there are many examples on the Web where this principle is not followed.) The other good idea was linking related data in documents. For example, this allowed us to create a hyperlink requirements traceability and verification matrix.

One person created the ISEE in about 80 hours using a freeware HTML editor. It was hosted on our LAN server, which, at the time, was a Compaq Proliant 1000 running Novell NetWare 3.1.2.

Next Implementation

The next attempt at implementing an Intranet site started with the format (at that time) of the TACCSF Web page and extended it to provide project information, information of general interest, and on-line administrative information.

This attempt also used a "what you see is what you get" (WYSIWYG) graphical editor (Netscape Navigator Gold); however, the WYSIWYG features of the editor were somewhat lacking, and a lot of custom page adjustments were required.

Another problem with this implementation was the lack of user involvement and requirements input, which meant that the system was never widely used.

The second attempt took about 40 hours to create and was also hosted on the Compaq server running Novell NetWare.

Current Efforts

We currently are experimenting with an Intranet implementation that will encompass the entire range of our daily operations and incorporate most of the elements that were present in previous attempts. This Intranet is intended to be more than a mere copy of the Web page or a source of project information.

One of the more important objectives of our current implementation is to increase usage of the Intranet. To accomplish this, it is essential to keep the data referenced in the Intranet pages current and to make the system easy to use and update. Fortunately, improvements in Web authoring tools and network operating system support for Intranets, along with increasing comfort levels with the Web browsing paradigm, make this objective easier to accomplish.

The organization of the current Intranet is being improved by designing the site map in advance (similar to doing a file plan for paper records) and assigning an "owner" for each of the sections of the Intranet site.

The latest Intranet sites were authored by one person at a time, with each iteration taking between eight and 16 hours. This illustrates how much authoring tools have improved since our first Intranet attempts. The latest iterations were created with AOLPress or Microsoft FrontPage and are hosted on a pair of Dell PowerEdge 2100s running Windows NT 4.0.

Future Plans

Once the current Intranet efforts are complete, we plan to start work on more interactive content, such as Java database connectivity, interactive forms, and on-line scheduling of conference rooms and test-bed resources. Other possibilities for growth are collaborative design and reviews with off-site customers and password-protected Internet access to the Intranet.

Our preference is to host future versions of the Intranet on its own server that runs Windows NT Intranet Information Server.

Selecting Tools

The quality and affordability of Intranet tools has improved considerably since we first started experimenting with an Intranet site. The text-oriented Web page editors in common use then have been replaced by graphical tools; network operating systems now routinely bundle Intranet capabilities as part of the package, and highly capable browsers are freely available.

The current range and quality of Web development tools is impressive. Most of the popular office suites have the capability to save documents in HTML format, and the appearance of these documents has improved greatly. Of course, to develop a comprehensive site and not just an HTML page requires a Web development environment. There are a large number of Web development tools available at a reasonable price, either stand-alone (Microsoft's FrontPage) or as part of a browser package (Netscape's Communicator). One tool that is worth consideration is AOLPress, which is a free package available at http://www.aolpress.com. There is also a free server package that complements AOLPress. There are better tools available; however, until you have more experience with Intranet development and can better determine your development environment needs, AOLPress can help you get your Intranet project started.

Lessons Learned

The most important Intranet development lesson learned is that your Intranet must be easy to use. This requires taking time to lay out the site in advance to ensure that the format and organization is logical and aesthetically pleasing.

You must also ensure that the Intranet users feel they have a voice in its development and use. If people feel the Intranet is being forced on them and are not given an opportunity to express their opinions about its design, they are not likely to use it.

On a similar note, it is important to identify "owners" for the sections of the site. We approached this by assigning a portion of the site to each of the directorates in the organization, and the directors assigned a person to maintain their section of the Intranet.

Once the site is established, maintenance is much simpler; however, it is important to realize that developing an Intranet is a resource-intensive undertaking. You need to take the time to evaluate why you want to establish an Intranet and how your organization expects to benefit from it. Then you can determine if the expected benefits will outweigh the effort required to implement your Intranet site.

If you follow these recommendations, you should be able to set up a successful Intranet site for your organization, big or small. ◆

About the Author

Capt. Dan Stormont is the chief of project engineering at TACCSF at Kirtland Air Force Base, N.M. Prior to TACCSF, he was the computer engineer for the B-1B Conventional Mission Upgrade Program at Wright-Patterson Air Force Base, Ohio. He has an associate's degree in electronics engineering technology from the Community College of the Air Force, a bachelor's degree in computer engineering from the University of Arizona, and is pursuing a master's degree in computer engineering at the University of New Mexico. He is an active member of the Association for Computing Machinery and the Institute of Electrical and Electronics Engineers Computer Society.

Det 4, 505 CCEG/ENP
1655 First Street S.E.
Kirtland AFB, NM 87117-5617
Voice: 505-846-0539 DSN 246-0539
Fax: DSN 246-1486
E-mail: dstorm@taccsf.kirtland.af.mil

Coming Events

Software Quality Through Robust Testing

Dates: May 21, 1998
Location: Eatontown, N.J.
Subject: Year 2000 fixes, reduce testing time and cost, improve coverage, and find defects early.
Contact: Madhav Phadke, Voice: 732-577-2878; Fax: 732-577-2879; E-mail: Madhav_Phadke@compuserve.com

7th IEEE North Atlantic Test Workshop

Dates: May 28-29, 1998
Location: West Greenwich, R.I.
Subject: Issues for the 21st Century: higher quality, more economical, and more efficient testing methodologies and designs.
Sponsor: IEEE Computer Society, Test Technology Technical Committee, University of Rhode Island
Contact: Jim Monzel, Voice: 802-769-6428; Fax: 802-769-7509, E-mail: jmonzel@vnet.ibm.com

Effective Methods of Defect Detection and Defect Prevention

Dates: June 2-4, 1998
Location: Seattle
Subject: Software quality decomposed into defect detection and defect prevention.
Sponsor: Quality Assurance Institute
Contact: Voice: 407-363-1111; Fax: 407-363-1112; Internet: http://www.qaiusa.com

5th International Conference on Software Reuse

Dates: June 2-5, 1998
Location: Victoria, British Columbia
Sponsor: IEEE Computer Society in cooperation with Association for Computing Machinery
Contact: Jeffrey S. Poulin, program co-chairman; Voice: 607-751-6899; Fax: 607-751-6025; E-mail: Jeffrey.Poulin@lmco.com

Second Workshop on Software Architectures in Product Line Acquisitions

Dates: June 8-10, 1998
Location: Hawthorne Hotel, Salem, Mass.
Subject: Applying software architecture technology to the acquisition of software-intensive product lines. Based on experiences, working groups will make recommendations to move to an architecture-based product line acquisition approach.
Contact: Lt. Col. Gene Glasser, E-mail: glassere@issc.belvoir.army.mil

15th International Conference on Testing Computer Software

Dates: June 8-12, 1998
Location: Washington, D.C.
Subject: "Testing Under Pressure," with emphasis on management strategies.
Sponsor: U.S. Professional Development Institute
Contact: Voice: 301-270-1033; Fax: 301-270-1040; E-mail: admin@uspdi.org; Internet: http://www.uspdi.org

4th Joint Avionics, Weapons, and Systems; Support, Software, and Simulation (JAWS S^3) Symposium and Exhibition

Dates: June 15-19, 1998
Location: Riviera Hotel, Las Vegas, Nev.
Subject: "Meeting the Technology Needs of the War Fighter in the Year 2000 and Beyond"
Sponsors: Director, Test, Systems Engineering and Evaluation, Office of the Undersecretary of Defense; U.S. Navy, Avionics Department; U.S. Air Force, Embedded Computer Resources Support Improvement Program; JAWS S^3 Working Group
Contact: http://jawswg.org

An Overview of the Extensible Markup Language and Related Content-Management Technologies

Greg Meyer
Nichols Research Corporation

The HyperText Markup Language is the most common document format encountered on the World Wide Web but is limited to presentation control. Several emerging technologies such as the Extensible Markup Language are currently being developed that promise dramatically enhanced content management on the Web. This article introduces these technologies and presents issues to consider when implementing them.

The vast majority of Web documents are created and presented in the HyperText Markup Language (HTML). HTML is well suited for hypertext linking and the display of small, relatively simple documents. HTML is an application of the Standard Generalized Markup Language (SGML), defined in ISO 8879:1986. SGML is a metalanguage designed to define document formats. SGML allows documents to describe their own grammar, which is implemented as a set of tags and the structural relationship that these tags represent. Although HTML also provides for a tag set that makes it easy to build Web documents, the HTML tag set is comparatively small and cannot be extended. This lack of flexibility limits HTML's ability to address extensibility, structure, and validation [1]:

- Extensibility: HTML does not allow users to specify their own tags or attributes, which limits their ability to parameterize or semantically qualify their data.
- Structure: HTML does not support the specification of deep structures needed to represent database schemas or object-oriented hierarchies.
- Validation: HTML does not support the kind of language specification that allows consuming applications to check data for structural validity.

At the other end of the functionality spectrum is SGML. SGML allows for the extensibility, structure, and validation that are missing in HTML. With SGML, document formats can be defined, and extremely large document repositories can be managed. However, SGML implementations are expensive, and SGML provides many features that are either unnecessary for Web publishing or require a large effort to implement in a Web environment.

Enter the Extensible Markup Language, (XML). As reported by *Time Magazine* in November 1997, "Doing business on the net is hard because the underlying software is so dumb. XML will fix that." [2] To put it just as eloquently, Bill Gates, chief executive officer of Microsoft, stated that "XML is a breakthrough technology."

Procedural and Generalized Markup

To understand XML and its impact on the Web, a brief introduction to generalized markup is necessary. Markup in electronic documents is the codes embedded in a document text that store the information required for electronic processing. Common examples of document markup include font family and font size.

Markup that represents a procedure for output devices is often referred to as procedural output. For example, when we use a word processor we choose fonts, boldness, and location of text on the page. By marking a word bold, we have defined a procedure that is carried out by an output device: When we view the document on a computer monitor, the word appears bold; when we print the document, the printer prints the word in bold. Although procedural markup can be valuable if all that concerns us is presentation, it has several limitations:

- Procedural markup does nothing to maintain information about the document structure—it is based on the assumption that document structure is directly related to document appearance; only document formatting is recorded and all structure is lost. For example, both quotations and emphasized words may be italicized, even though a quotation has a different function than an emphasis.
- Procedural markup is time-consuming and requires a significant amount of operator training. For example, the documentation for a large software development project may contain thousands of pages, and each of these pages might adhere to a standard formatting convention—the effort to ensure this adherence can be extremely costly.
- Procedural markup is inflexible. When a change to a formatting convention is applied, it requires the manual change of all elements in the document that are affected. In addition, the formatting codes are system dependent: One system may have a particular typeface that another system lacks.

Unlike procedural markup, generalized markup is not concerned with formatting. A Generalized Markup Language (GML) requires two characteristics from the markup:

- Markup should not describe the processing to be performed on the

document; rather, markup should describe the document's structure. This descriptive markup needs to be done only once and will apply to all future processing.
- Markup should be formally defined. With this formal definition of markup, external programs can be used to process the document.

In a GML, a tag is attached to text elements, and formatting rules are associated with these tags. A formatter processes the text and produces a document in a format that is suitable for the output device. The advantages that a GML has over a procedural markup process include the following:
- Generalized markup describes document structure. Meaningful names can be given to tags, such as <PARAGRAPH> to represent a paragraph, and <SURNAME> to represent a person's last name. This application of meaningful names to tags allows the automatic processing of the document, such as the compilation of an index of tagged words.
- Generalized markup allows for much flexibility. To change the appearance of the document, it is necessary only to modify an external procedure that processes the document. This single modification will suffice for all occurrences of the appearance change, and the labor involved with hundreds or thousands of manual changes can be avoided.

XML

XML is a coding system that allows any type of information to be delivered across the Web. Like HTML, the heritage of XML is in SGML. In fact, like HTML, XML is often considered an SGML application (technically, HTML is an SGML application, whereas XML is an SGML profile).

The XML specification was developed by a group of SGML industry leaders and the World Wide Web Consortium (W3C) in 1996, with Jon Bosak of Sun Microsystems as the acting chairman. The goal of the XML Working Group (originally known as the SGML Editorial Review Board) was to develop a markup language that had the functionality of SGML but could be effectively presented on the Web. The initial working draft was completed in late 1996 and became a W3C Proposed Recommendation Dec. 8, 1997 and a W3C Recommendation Feb. 10, 1998. The recommendation (REC-xml-19980210) can be found on the W3C site at http://www.w3.org/TR/1998/REC-xml-19980210.html.

The recommendation outlines the design goals for XML [3]:
- XML shall be straightforwardly usable over the Internet.
- XML shall support a wide variety of applications.
- XML shall be compatible with SGML.
- It shall be easy to write programs that process XML documents.
- The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
- XML documents should be human-legible and reasonably clear.
- The XML design should be prepared quickly.
- The design of XML shall be formal and concise.
- XML documents shall be easy to create.
- Terseness in XML markup is of minimal importance.

The results of the XML Working Group is a GML that allows the creation of new tag sets, instead of being forced to use the minimal tag set available in HTML. More important, XML allows documents to be self-describing and provides for the validation of documents.
- XML documents are self-describing in that they can contain header information known as a Document Type Definition (DTD). The DTD describes the structural rules that the markup in the document is to follow, declares internal and external resources that form part of the document or might be required within the document, and lists non-XML resources that are found in the document for which external helper applications are required. This DTD is instrumental in the successful application of XML processing software.
- XML and a DTD enables a document to be validated by describing a rule set to which that the document must adhere. (It is not necessary for an XML document to contain a DTD—XML documents without a DTD are considered *well formed* but not *valid*. A *well-formed* document adheres to a standard set of rules such as a requirement that each opening tag is accompanied by a closing tag.)

So what is the end result of an XML document? It can be summarized as
- A document that "understands itself" – header information that specifies which elements are allowed and the properties of these elements.
- A document with a browseable and searchable structure – the refusal to allow the exclusion of necessary markup tags allows XML documents to be accessed by XML-aware tools.

The best way to appreciate XML is to look at an example of XML code. In this example, imagine that a company sells automobile parts on line. Marketing descriptions of the products are written in HTML, but names and addresses of customers, prices, and discounts are formatted with XML. Following is the information that describes a customer.

```
<CUSTOMER-DETAILS>
  <NAME>American Wholesale Auto Parts</NAME>
  <ADDRESS>
     <STREET>1234 Maple Drive</STREET>
     <CITY>Grayson</CITY>
     <STATE>Colorado</STATE>
     <ZIP-CODE>80113</ZIP-CODE>
  </ADDRESS>
</CUSTOMER-DETAILS>
```

The XML tags such as <STREET> and </STREET> give meaning to the text "1234 Maple Drive." Its simple syntax is easy to process by machine and has the attraction of remaining understandable to humans.

Related Technologies

As with any other emerging technology, XML brings along with it a host of related technologies. Two of the most impor-

tant of these technologies are the XML linking mechanism and XML style sheets.

XML Linking Language (XLink)
(http://www.w3.org/TR/WD-xml-link.html)

XML linking is defined by the XLink specification as "a simple set of constructs that may be inserted into XML documents to describe links between objects and to support addressing into the internal structures of XML documents. It is a goal to use the power of XML to create a structure that can describe the simple unidirectional hyperlinks of today's HTML as well as more sophisticated multiended, typed, self-describing links." [4] XLink allows specification of which elements in a document are to be interpreted as links and the specific nature of these links. For example, a default link behavior can be defined that requires the user to take a specific action before anything is done with the link. XLink also introduces extended links into Web documents. Extended links can point to any number of targets and can also be bidirectional and multidirectional.

Extensible Style Language (XSL)
(http://www.w3.org/TR/NOTE-XSL.html)

XML style sheets are defined by the XSL specification as "the deliverable for Phase III of the SGML, XML, and Structured Document Interchange Activity of the W3C." [5] The charter for this activity specifies the use of ISO/IEC 10179 Document Style Semantics and Specification Language (DSSSL) for the style-sheet language component. XSL is based on DSSSL and is a style-sheet language designed for the Web community. It provides functionality beyond HTML's Cascading Style Sheets (CSS) such as element reordering. It is expected that CSS will be used to display simply structured XML documents, and XSL will be used where more powerful formatting capabilities are required or for formatting highly structured information such as XML-structured data or XML documents that contain structured data.

Capabilities provided by XSL allow the

- formatting of source elements based on ancestry and descendency, position, and uniqueness.
- creation of formatting constructs, including generated text and graphics.
- definition of reusable formatting macros.
- writing of direction-independent style sheets.
- creation of an extensible set of formatting objects.

A few other XML-related technologies include the following:

Resource Description Framework (RDF)
(http://www.w3.org/Metadata/RDF/)

RDF may prove to be one of XML's most important applications. RDF allows applications to describe new data fields and classes—defining relationships between XML data that might otherwise be left undefined. For example, RDF can be used for bookmarks, user preferences, and a host of other information not directly related to an XML document. This application is a prominent use of metadata (data about other data). RDF will enable enhanced search engines, descriptive relationships between content within a single Web site or between different Web sites, and content ratings for privacy and child protection.

Channel Definition Format (CDF)
(http://pushconcepts.com/microsoft.htm)

CDF provides the ability to author content once for publishing via many different vehicles using push, pull, and static mechanisms. CDF depends on XML for its declarative syntax.

Open Software Description Format (OSD)
(http://www.w3.org/TR/NOTE-OSD.html)

OSD uses unique XML tags to describe software components, including their versions, underlying structure, relationships to other components, and dependencies. It can describe and reference platform native code. Software packages that are described using OSD can be delivered automatically using push technology, allowing for simplified software upgrades and avoiding cross-platform installation complexities.

Open Financial Exchange (OFX)
(http://www.ofx.net)

OFX is a framework for exchanging financial data and instructions among financial institutions and their customers.

XML/Electronic Data Interchange (XML/EDI)
(http://www.geocities.com/WallStreet/Floor/5815)

XML/EDI provides a standard framework to describe different types of data such as shipping invoices and health-care claims. XML/EDI allows information in these various types of data to be searched, decoded, manipulated, and displayed consistently and correctly by implementing EDI dictionaries.

Implementing Content Management Technologies

As an emerging technology, XML has yet to garner widespread industry tool support. This, however, is sure to change in the near future. At a minimum, XML implementation requires an XML or ASCII text editor, an XML parser, and an XML viewer.

Any ASCII text editor can be used to author XML documents. However, there are a few XML-specific authoring tools that make the authoring process significantly easier. XML can be parsed using several tools.

A few of the growing bin of XML-specific software include

- Jumbo (by Peter Murray-Rust) – a set of Java classes designed for viewing XML applications (http://ala.vsms.nottingham.ac.uk/vsms/java/jumbo).
- DataChannel XML Development Kit (by DataChannel) – an enterprise development tool to integrate databases, legacy systems, and business-to-business transactions over the Web using XML (http://www.datachannel.com).
- Lark (by Tim Bray) – an XML processor written in Java (http://www.textuality.com/Lark).
- Copernican XML Developer's Toolkit (by Copernican Solutions) – a toolkit that provides for the checking, valida-

tion, loading, and access of XML documents (http://www.copsol.com/products/xdk/XDK).
- Internet Explorer 4.0 and MXSML (both by Microsoft) – MXSML is an XML parser written in Java, whereas Internet Explorer contains the first public implementation of an XML engine within a Web browser (http://www.microsoft.com/workshop/author/xml/parser).
- TclXML (by Steve Ball) – a Tcl add-on that allows the parsing of XML documents and DTDs (http://tcltk.anu.edu.au/XML).
- XML Styler (by Arbortext) – an XML style sheet editor (http://www.arbortext.com).
- FrameMaker (by Adobe) – a comprehensive document-authoring suite that is XML enabled (http://www.adobe.com).

Closing Thoughts

A final but important question to think about when considering the implementation of XML is the large base of HTML documents that currently exist. Does the advent of XML portend the demise of HTML? Probably not. In most cases, developing XML applications will not be cost effective. HTML is an application that works without modification, there is an incredibly large base of authoring software for creating HTML pages, and a strong industry exists that provides search and retrieval tools for HTML. Information technology is now, and will probably be for many years, concerned primarily with delivery of static information. HTML will probably continue to provide the ideal solution for the bulk of information delivery across the Web for several years.

However, if intelligent information management across the Web is required, XML is a viable, if not dominant, solution. With self-describing and validating mechanisms, browseable and searchable document structures, sophisticated linking, and incredibly flexible presentation support, XML is ideally situated to leverage the information in Web-based documents. ◆

About the Author

Greg Meyer is a principal consultant in the Enterprise Solutions Business Unit of Nichols Research Corporation. He has over 13 years experience in complex systems integration, Web site development, imaging, document management, and collaborative computing. He is a certified document imaging architect and a master of information technology in the Association for Information and Image Management. His professional interests include document management, Web-based collaborative computing, and markup languages.

Nichols Research Corporation
10260 Old Columbia Road
Columbia, MD 21046-1707
Voice: 410-290-9500, ext. 103
Fax: 410-290-7012
E-mail: meyerg@nichols.com
Internet: http://www.nichols.com

References

1. Bosak, Jon, "XML, Java, and the Future of the Web," http://sunsite.unc.edu/pub/sun-info/standards/xml/whyxmlapps.html, March 10, 1997.
2. Krantz, Michael, "Keeping Tabs Online," *Time Magazine*, Nov. 10, 1997.
3. Bray, Tim, Jean Paoli, and C.M. Sperberg-McQueen, "World Wide Web Consortium Recommendation: Extensible Markup Language (XML) 1.0," http://www.w3.org/TR/PR-xml-971208.html, Dec. 8, 1997.
4. Bray, Tim and Steve DeRose. "World Wide Web Consortium Working Draft: Extensible Markup Language (XML): Part 2. Linking," http://www.w3.org/TR/WD-xml-link-970731.html, July 31, 1997.
5. Adler, Sharon, et al., "World Wide Web Consortium Note: A Proposal for XSL," http://www.w3.org/TR/NOTE-XSL-970910.html, Aug. 27, 1997.

Additional Reading

1. Light, Richard and Tim Bray, *Presenting XML*, Sams Publishing, Indianapolis, Ind., 1997.

Web Addition

This article can be found in its entirety on the Software Technology Support Center Web site at http://www.stsc.hill.af.mil/CrossTalk/crostalk.html. Go to the "Web Addition" section of the table of contents.

Organizations, Operations, and Officers

Databases Supporting NATO's Operations in Bosnia

Joseph Arsenault
Canuck Consultants

Long before the first allied solider crossed the Sava River into Bosnia or the first air force crew person landed at Sarajevo Airport or the first maritime sailor debarked at the Split, Croatia seaport, databases were being used to plan, organize, track, and deploy allied forces. This article presents a review of databases that support NATO's Implementation Force and Stabilization Force operations in Bosnia.

Web 103: The Lazy Person's Guide to Intranets

Maj. Dale Long
U.S. Air Force

An Intranet is a computer-mediated communications infrastructure based on Internet communication and content standards with access limited to clients in a particular institution or community. Many organizations have an Intranet, are installing an Intranet, or are considering an Intranet. This article covers some of the basics of Intranets and how to make them more than just expensive window dressing for your desktop systems.

INTRANET, IN´TRA NET´, n. 1 A computer-mediated communications infrastructure based on Internet communication and content standards with access limited to clients in a particular institution or community. 2 A cancerous proliferation of hypertext markup language files. 3 Organizational Excedrin Headache No. 8879.

Chances are your organization has an Intranet, is installing an Intranet, or is thinking about installing an Intranet. This article will cover some of the basics of Intranets and how to make them more than just expensive window dressing for your desktop systems.

Intranet Basics

As noted in the definition above, Intranets are based on Internet communication and content standards. What distinguishes an Intranet from the World Wide Web is that access to information published on an Intranet is restricted, usually through the use of local area networks protected by fire walls. They are not limited to a single physical network and may span many networks at various locations.

There are four key roles associated with Intranets: users, authors, publishers, and brokers. People may assume any of them in the course of their work.

User Issues

Users benefit from the content of the Intranet. This is the group for whom we must design, as they are the reason the Intranet exists. While this is an easy group to define, it is rarely an easy group to satisfy. Users will have a wide range of needs and capabilities. A system simple enough for the first 80 percent of the population will rarely satisfy the top 20 percent. A system designed for the top 20 percent may be too complex or frustrating for the rest of the world.

Authors

Authors create content and structure. Content is the information itself, usually presented as HTML files, word processing documents, spreadsheets, or database reports. Structure defines the information relationships within and between various forms of content.

In traditional publishing, content and structure are most often absolutely linked—the author has complete control over the linear nature of the information and how it is presented.

However, content changes in a hyperlinked model. Instead of republishing information, authors can merely link to other documents. The less duplication of information, the less maintenance required to synchronize various information sources. Linked, reusable modules of information combined with nonlinear, cross-referential structures will radically transform how we use and maintain information.

Structure has also changed because of hypertext publishing. Hyperlinks let users pick and choose the information they wish to see and in what order. Good hypertext authors will develop structure that helps users determine which information is most valuable for their current need instead of locking them into the more traditional "I'm the author; I know best" linear model.

As a rule of thumb, design simple information structures. Even Nobel Prize laureates probably would not appreciate having to hunt through a complex, arcane set of hyperlinks to find what they need.

Publish or Perish

Publishers make information available. They manage, coordinate, and communicate content in (I hope) predictable and efficient ways. They determine which content is most appropriate for their organization, what structures through which to present it, and how to manage its lifecycle.

In the digital environment, publishing no longer means presenting a static representation of information. Electronic publishers have a far greater reach and scope because of the variety of content available and the speed at which they can disseminate it.

However, publishing is a part of our traditional organizational bureaucracy that we established, developed, and maintained to deal primarily with paper-based information. Unfortunately, most of our bureaucracies and organizational processes are little more than mechanisms to move information on paper, not to manage the information itself.

The digital environment presents new opportunities and challenges in this regard. Digital mechanisms, like E-mail and Intranets, threaten to break down those established bureaucracies. Naturally, they resist, the result of which is that Intranet publishers frequently end up merely transplanting old paper process "sacred cows" into their digital enterprises.

If you are a publisher, resist this. When moving operations from paper to digital environments, rigorously examine every information process you own. If you can automate a task, e.g., assigning sequential numbers, do it. If you can bypass information choke points without bringing your organization to a standstill from information overload, do it.

Shoot some sacred cows. They usually make the best hamburger.

This article is based on "The Lazy Person's Guide to Intranets," Chips, January 1998. For reprint permission, contact Chips at http://www.chips.navy.mil.

Publishers need different information structures than users to help them manage information content. The most important of these structures is an information map that describes their content, where it resides in the system of information, its relationship to other content, and the rules for access. Without a good map, an Intranet can quickly become an indecipherable Gordian Knot of bad links, obsolete information, and other useless junk.

Brokers

Brokers help us find information. This is true in both the paper and the digital environments. Understanding how brokers work will become increasingly important as we adopt Web technology, which allows prolific, independent creation of information well beyond our capacity to find and use it for specific needs.

Commonly used paper information brokers are the telephone book, bibliographies, and indexes. Librarians, researchers, and political pollsters broker information. A good information broker is ubiquitous. If it works well, we hardly notice it. Polls, for example, are everywhere. But do we even notice the mechanisms used to produce all that information we are bombarded with daily?

Where Web technology is having its greatest effect is on brokering access to information. A crucial feature of on-line brokers is that they deliver information access pathways instead of just documents. Digital brokers can search for and screen vast amounts of information in a relatively short period.

This radically alters the focus of the information broker. A paper document might come as 50 pages of text and graphics. On the Web, however, a document may be a single page with content and hyperlinks. The authors probably created those documents by editing together information from other documents using brokers.

With more advanced Web technology, we should be able to build brokers that understand users' decision processes and structure access paths to appropriate content to better support those decisions. We are not talking about another application like PointCast, here. The goal should be to develop autonomous intelligent agents that search for what we need across the entire infosphere, not just download simple, pre-programmed content.

Adapt and Evolve

Military organizations have been struggling with some increasingly complex information management issues for decades. A major issue is the seemingly constant increase in the amount of information we have to deal with every day. If we do not find some way to harness and control our ballooning information stores, we will eventually reach a point where we are no longer able to make effective use of everything we gather or produce. If we have not passed that point already.

The most basic visible effect of this is called a surface-to-volume ratio. Imagine the sum total of our organizational information resources as a sphere. The surface area represents the amount of information we have to deal with every day. The volume represents the infrastructure (people, equipment, and process) required to support our information production and consumption.

In a normal sphere, as the surface area increases, the volume also increases to support the larger surface area. To do this, the volume must increase twice as fast as the surface area.

To use a more common metaphor, our infospheres are more like basketballs. The larger the ball, the greater the surface area and the more infrastructure support (air pressure) it requires to stay inflated and useful.

Increases in infrastructure also usually increase inertia. As a system or organization grows, it acquires mass. More mass means more effort required to change anything, as "a body at rest tends to stay at rest."

If we build a large, complex infrastructure, we will require more coordination to make decisions, have more layers through which information passes, and need more people to manage the whole system. Decision makers at the center of the sphere must deal with an increasing number of competing inputs, which can be confusing at best and debilitating at worst.

Our surface area, the information we need to do business, must have the volume to support it. Without that volume, the surface of our sphere will collapse, much like a basketball without enough air.

And our information surface area is increasing every day.

Pump Up the Infrastructure

Earlier, I mentioned three components of our information infrastructure: people, equipment, and process. Let us examine some of our options for building more support for our information needs:

Option No. 1: Hire more people.

Chances of this are just about nil nowadays. But we can always dream.

Option No. 2: Add more equipment.

We are doing this now, but you cannot solve a business problem just by throwing technology at it. While extensive computer and network infrastructures are becoming an essential part of military life, they also are one of the primary reasons we are becoming overwhelmed with information. We are now dealing with volumes of information well in excess of what our current organizational systems were originally designed to handle.

Option No. 3: Change our processes.

This is where we must focus if we intend to survive and prosper. It sounds like a simple answer, but it is not. Processes acquire inertia, too, and it usually requires a significant amount of effort to make any substantive change to them.

Although technology cannot solve our problems by itself, we absolutely do need more advanced hardware and software to support these essential process changes. However, we must ensure that we are applying the right technology to the process and that we are changing the process in the right way.

A famous example of process paradigm shift (and lack thereof) occurred when the photocopier was first invented. Two of the biggest manufacturers in the office automation world of the day passed on buying the rights to the technology. Their reasoning?

Photocopying was too expensive to replace carbon paper, which was, at that time, the standard way to make copies.

The people who accepted that rationale were stuck inside a fairly narrow box in which carbon paper was the main way to make copies. However, xerography was not intended to mirror the process of producing copies at creation but to allow reproduction after a document had been created.

We know all about how ignorant they were back in the dark ages when computers still used vacuum tubes. But how many of our opportunities today look the same way to us?

So, we need some fairly radical process change fueled by an infusion of technology that supports organizational and personal use of ever-growing volumes of information. And we have to do with fewer people every year.

There are no simple answers to this problem, but we can always hope that someone else will invent the "next big thing" and solve our problem for us.

However, hope is not a strategy.

We already have the technology. We just have to find the will to employ it.

Why an Intranet?

Intranets allow organizations to restructure their information operations to allow distributed, rather than central, management of information and decision making. This is a significant change in thinking in some circles.

Organizations succeed by leveraging the benefits of coordinated activities. In highly successful organizations, members at each level become self-regulating and standardized; they share the common organizational goals and purposes, and any energy that was previously devoted to regulation or control may now be redirected to production.

Complex organizations composed of self-regulating subsystems should be more responsive than monolithic organizations of similar size because the smaller, semiautonomous parts will react faster to the same stimuli.

It is something like boiling a potato.

Divide and Mash

Take two potatoes of equal size. Cut one into one-inch chunks, and leave the other whole. If you drop them both into boiling water, the small chunks will cook much faster, which means your potato reaches the desired end state much faster. Why do they cook faster? Because they have a much smaller surface-to-volume ratio.

The center of the potato is much closer to the surface in the small chunk than it is in the large one, so it cooks faster. In a distributed organization, the decision-making elements, the centers of the chunks of the organization, are closer to the information they need, have less infrastructure to wade through, and can react faster in coordination with other organizational components to get the job done.

The processes we use for communication and coordination among our self-regulating organizational components will determine our ultimate levels of performance. Our current processes of communication and coordination, however, are still anchored in paper-based approaches, even where we have transferred the information to digital media.

Unfortunately, we have probably reached the limits of size and complexity that we can support with a paper-based communications infrastructure. It is no longer just about building large computer networks; it is managing organizational information resources. We must build comprehensive information infrastructures that will allow us to create more agile and more responsive organizations.

The Role of the Intranet

Intranets, along with E-mail, databases, and other technological marvels, will help redefine what we consider high-performance organizations. Each has its uses.

E-mail is great for point-to-point communications. Databases crunch critical mission data and presents structured results. Other components, like imaging and work-flow, have their places, too. Where does an Intranet fit in with the information infrastructure?

First, it is a private publishing medium. The first thing that most organizations do is build their Intranet around their organizational hierarchy and fill the server with mission and vision statements. Although this is not a particularly productive use of hard-drive space, it does not hurt and usually generates a "warm fuzzy" effect for everyone who sees their name or their organization's name up in hyperlights.

The first real value you accrue from an Intranet is relatively universal access to functional information that crosses traditional organizational boundaries. This will usually be the second set of pages authors generate. You may, if you wish, reference your entire functional knowledge store through both the Intranet's formal structure (home pages) and an indexed search. This facilitates a whole new level of information sharing between organizational subunits and individuals.

Note that I said Intranets are private, not secure. While all this access is good from a sharing standpoint, it also means that a lot more people may now easily replicate and distribute whatever is out there. There are still some types of information that, while we want to facilitate sharing, we do not want to share with everyone.

One of the greatest barriers to effective information sharing is not that people are not allowed to see information, but that they do not know it exists. However, there is a happy medium. You can publish pages that contain links to sensitive files, but further restrict access to those files at the system level.

Home pages and index searches should show all the documents that are available through the system so people know what they may be missing. But control of the individual files should still belong to the authors and publishers responsible for that information. If files need protection, protect them. But do not miss out on the benefits of Intranet access by withholding potentially important documents from the common repository.

Intranets also can function as an access shell as part of a three-tiered computing architecture.

The classic two-tier client-server computing model separates presentation and calculation from data. The data sits on the server, and the client performs the work. This was the original model for most client-server networking.

However, there were some problems with enterprise-wide two-tier architectures, particularly when you tried upgrading an application or distributing load

processing. Three-tier architectures, which separate by presentation, business operation, and data, are more common now.

In a three-tier system, the bottom layer is the data. The middle layer holds all the network applications that work with the data: databases, work-flow engines, indexing systems, imaging systems, etc.

The top layer, presentation, is what sits on the client's desktop. In the past, much of the presentation layer has been monopolized by proprietary interfaces dedicated to a limited set of functions. If you wanted to talk to the database, you had to use one interface. If you wanted to read word processing documents, you had to use a word processor. Every data type has an associated application, and the only overall representation of your information repository was through a file manager or viewer of some type.

Now, however, one of the most prolific client applications is our Web browser.

This is a good thing for three reasons:

First, Web browsers are fairly universal. Even with all the contention about the next HyperText Markup Language (HTML) standard between vendors, you can often bring up someone else's HTML page. In addition, browsers are rapidly gaining the ability to read non-HTML files via plug-ins or associated applications. Views of the information repository are no longer limited to how file structures are arranged thanks to Web publishing mechanisms.

Second, access through a Web browser is as simple as clicking on a hyperlink. The learning curve is not steep for most users.

Finally, you can administer all the information in the Intranet either centrally or remotely. Webmasters can help less accomplished authors and publishers get their information out. More experienced authors and publishers can manage their own chunks of the system independently as part of a distributed but coordinated component of the total information infrastructure.

There are other applications that can provide this type of functionality, but most of them are expensive, proprietary, and dying out because of the ease of use of browser technology and the relative simplicity and power of HTML. Good ideas can spread rapidly on an Intranet, too. One author's great HTML can be immediately copied and replicated throughout an organization in a relatively short period.

Sharing is good.

Rules to Live By

This last section covers a few rules you should establish for your Intranets.

First, strike a balance between organizational and functional content areas. Every organization should probably have its own set of "we love us" pages. However, that is not where you will get a substantial return on your investment.

The public library card catalogue, still one of the most functional retrieval systems ever devised, sorts by author, title, and subject. Organizational pages are our equivalent of an author search. If you know who published it, you can find your information. But if that is all you have, you only have one-third of the brokering capability your users need.

Index searching is a valuable tool. Spend time on training users how to conduct Boolean searches. It is well worth it to the entire organization.

Functional managers, including those running ad hoc or temporary groups, must create a presence for their content on the Intranet. Personnel news, social events, organizational policy, and a host of other information should not be buried deep in some branch's organizational page. Put what is important to the entire organization at the highest levels, regardless of where the author sits in the formal hierarchy.

Second, do not hold back on content. Hard drives are inexpensive, and index engines are getting better every day. If we have good brokers on the system and we train people how to use them, we can exponentially increase the amount of useful information available to our organizations. Restrict access to what you need to, but do not hide something that exists unless its existence is supposed to be a secret.

Third, try to keep the junk and bandwidth hogs to a minimum. If a lot of your population is still using 486/33s, do not let authors stick huge, spinning, three-dimensional, animated, 1 megabyte graphics that take a full minute to load on their home pages. Graphics are good, but some are just gratuitous.

Fourth, enforce your standards. Few things can get out of control faster than an Intranet, particularly if all those distributed authors and publishers have different visions of how things should be. In my earlier endorsement of distributed, decentralized decision making, I mentioned coordinated activities, which means standards, standards, and standards.

Fifth, review all pages at least monthly for currency. Nothing debilitates an Intranet like link rot. Dispose of obsolete information, but make sure you have some provision to archive any electronic files that may qualify as federal record material.

Finally, if authors or publishers rename, delete, or move pages, they should create and maintain a notice page indicating what happened to the old page and provide a hypertext link to the new page. Keep these notices up for at least 30 days.

Final Words

> "Technology is a way of organizing the universe so that man doesn't have to experience it."
>
> — Max Frisch

If that is so, maybe Intranets can be a way of organizing our infospheres so we work less on information and more with it. ◆

About the Author

Maj. Dale Long is chief of Command Information Management, United States Strategic Command, Offutt Air Force Base, Neb. He has a master of science degree in information resource management from the Air Force Institute of Technology and a master of music degree from Northwestern University.

Chief, Command Information Management
USSTRATCOM/J073, Suite 1E5
Offutt AFB, NE 68113-6073
Voice: 402-294-2060 DSN 271-2060
Fax: 402-294-2318 DSN 271-2318
E-mail: longd@stratcom.af.mil

Impact of Interconnection Network Resources on Software Productivity, Reliability, and Maintainability

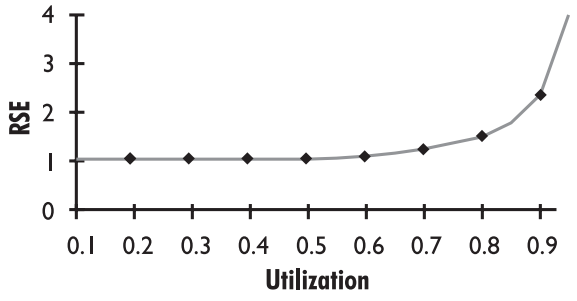
Wes C. Berseth and John A. Neff, *University of Colorado*
Anis Husain, *Defense Advanced Research Projects Agency*
Bill Wren, *Honeywell Technology Center*

Limitations in processing and memory resources are known to adversely affect software productivity. Our findings indicate that limitations in interconnection network resources affect software as much as or more than processing and memory constraints. This work forms a basis to extend existing software estimation models to describe real-time multiprocessor systems.

Software is of paramount importance to the U.S. military, providing advanced surveillance, intelligence, and weapons capabilities. However, software development projects are often over schedule and over budget, and the resulting software is delivered with an unacceptable number of defects. As a result, considerable concern has been expressed regarding the capability of current software engineering practices to enable the information dominance desired by the U.S. military [1]. With current trends in real-time embedded multiprocessor applications, these problems will become increasingly more difficult to alleviate. The demand to provide greater functionality while reducing cost and cycle time is increasing, and consequently, system design and development are becoming more complex [2, 3]. A greater portion of this functionality is being implemented in software, which is becoming more complex and difficult to develop and which comprises a greater proportion of the cost of an application [3, 4]. For large digital signal processing application software, costs are typically greater than hardware costs, often comprising 70 percent to 80 percent of the total cost, which can be several million dollars.

One strategy to alleviate these growing problems is system modeling in which the impact of the major factors that impact software cost, reliability, and maintenance are explicitly included. With an appropriately modeled system, simulation and trade-off analyses can be performed to optimize the system for cost while minimizing reliability and maintenance costs.

Figure 1. *Relation between relative software effort (RSE) and processor utilization as determined from the PRICE S software estimation model [5].*



	Hardware Costs	Software Costs	Total Cost	Development Time
Minimum Hardware Cost	$281,000 (11)	$2,360,000 (89)	$2,640,000	32 Months
Minimum Total Cost	$432,000 (32)	$911,200 (68)	$1,343,200	28 Months

Table 1. *Comparison of costs and development time between minimum hardware cost and minimum total cost scenarios. Percentages are in parentheses. Data taken from [6].*

Here we discuss the impact of hardware on software development and report our findings on the impact of a major multiprocessor hardware component—the interconnection network—on software cost, reliability, and maintenance. We believe this work provides a basis to extend current parametric cost-estimating models to describe real-time multiprocessor systems.

Processing and Memory Resources Impact Software Productivity and Quality

It is well known that limitations in processing and memory resources increase the effort required to develop software. Under these constraints, developers must deal directly with the operating system and hardware; therefore, detailed knowledge of the machine architecture is required. Coding in low-level languages is often required and high-level development tools cannot be used [5]. This effect is expressed in parametric cost-estimating models applied to software development, e.g., Constructive Cost Model (COCOMO), Revised Enhanced Version of Intermediate COCOMO (REVIC), PRICE S, and SEER-Software Estimating Model. For example, with the PRICE S model, when processor utilization—defined as the fraction of available hardware cycle time utilized—is below 0.5, software effort is not affected. As utilization is increased above 0.5, the required software effort increases nonlinearly and is 2.33 and 4.0 times higher relative to nonconstrained utilization when processor utilization is 0.9 and 0.95, respectively (Figure 1). The identical relation-

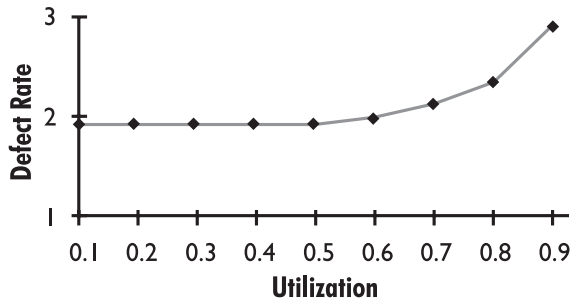
ship is given for memory utilization and software effort [5]. The defect rate, which affects the reliability of the system, increases in a similar fashion, increasing from 1.8 errors per thousand lines of code to 2.9 at processor utilization of 0.5 and 0.9, respectively (Figure 2).

Adding Processing and Memory Resources Can Decrease Software Costs

As part of the RASSP (rapid prototyping of application-specific signal processors) program, Jim Anderson of the Massachusetts Institute of Technology Lincoln Laboratory used parametric cost-estimation techniques to examine the effects of memory and processor constraints on the development costs of a synthetic aperture radar (SAR) processor [6]. For his model system, he chose an unmanned air vehicle (UAV) SAR benchmark developed at Lincoln Laboratory for the RASSP program. According to Anderson, hardware costs can be minimized by supplying enough processing (1 billion floating point operations per second) and memory resources to meet and not exceed application requirements. With commercial hardware, this can be realized with six Mercury MCV6 4 x 4m cards, each with four 40 megahertz Intel I860 processors and 16 megabytes of Dynamic Random Access Memory. In addition, a commercial back-plane-mounted crossbar switch, a Motorola MVME167 system controller card, and a custom radar interface card are required. The resulting processor will have a memory utilization of 86 percent and processor utilization of 88 percent for a total hardware cost of $281,000 (Table 1). According to Anderson, these requirements are not unusual for UAV applications where size, weight, and power must be minimized. However, the software costs and development time corresponding to the above memory and processor constraints are $2,360,000 and 32 months, respectively, as determined by the REVIC software cost estimating model. This cost was determined by estimating the code to be 8,750 uncommitted source lines of code, requiring 155 programmer-months, 152 programmer-hours per programmer-month, and $100 per programmer-hour. With this minimum hardware cost scenario, software development is 89 percent of the total development expense of $2,640,000.

Minimizing the total system cost can be achieved by adding enough hardware resources so that memory and processor utilization does not have an adverse affect on software cost. This occurs when both memory and processor utilization are below 50 percent (Figure 1). This is achieved by increasing the number of Mercury MCV6 cards from six to 11 with no change to the rest of the hardware. The result is an increase in hardware costs by a factor of 1.8, to $432,000, and a decrease in software costs by a factor of 2.59, to $911,200 (Table 1). The development time is also decreased from 32 to 28 months. Anderson's example shows how development cost can be dominated by software and that a greater investment in hardware can substantially reduce overall system development costs. The net result is a superior product at half the cost of the minimum hardware product.

Figure 2. *Typical relation between processor utilization and defect rate. Data courtesy of Jim Otte, PRICE Systems.*



Interconnection Network Resources Impact Software Productivity and Quality

Interconnection bandwidth[1] has been identified as directly impacting the ease of programming large supercomputers for high performance [7, 8]. Consequently, software development costs can be substantial due to the considerable effort required to obtain specific optimizations that are highly tuned to the particular distribution of data and machine [9]. This sentiment was supported by numerous personal communications with experts in the supercomputer and real-time embedded digital signal processing domains. Howard Shrobe has identified limited bandwidth as one of the "seven deadly sins" of software engineering [10]. Although it has been recognized that programming multiprocessors is more difficult when interconnection bandwidth is limited, this relation has not been quantified.

Bisection Bandwidth and Software Productivity

To quantify the relationship between bandwidth constraints and software productivity, estimates were obtained from experts experienced in the development of multiprocessor applications—seven experienced in real-time embedded signal processing and one in high-performance supercomputing. In addition, a questionnaire, developed at the University of Colorado, was used to collect additional information, e.g., impact on software quality and maintenance, and provide a check on model estimates. Eleven multiprocessor experts responded to the questionnaire, including seven of the above eight. We believe this was a suitable approach to determine the general relationship between bandwidth constraints and software productivity, thus reflecting an industry average. As a measure of bandwidth constraint, we use bisection bandwidth utilization (BBU), defined as the average fraction of available bisection bandwidth[2] that is used during data transfers. We believe this to be a useful measure of the difficulty encountered by a software developer, since bisection bandwidth is the critical bottleneck when performing global data transfers such as corner-turn[3] operations inherent in digital signal processing applications. Also, it is consistent with measures of hardware constraints used in existing parametric cost-estimation models, i.e., processor and memory utilization.

As a measure of software productivity, we define relative software effort (RSE) as the ratio of the effort required to de-

velop software relative to the effort required if bandwidth were not constrained. A relative measure was chosen to normalize data and enable pooling from a broad range of application parameters, such as size of application and programming language.

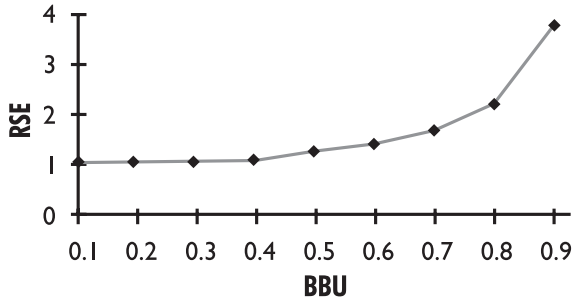
The relationship between BBU and RSE is given in Figure 3. RSE is not affected until BBU reaches 0.3, beyond which RSE increases nonlinearly, increasing to 3.8 at a BBU of 0.9. As BBU approaches 1.0, RSE becomes extremely high and the relationship is undefined, although in practice other factors likely become important. Some of the experts interviewed stated that dramatic increases in software costs initiate other decisions. For example, the program can be temporarily called to a halt while the organization waits until faster hardware becomes available or custom hardware is developed.

These results indicate that bisection bandwidth constraints have a more dramatic impact on software productivity than either memory or processor constraints. The adverse impact of bandwidth constraints sets in at a BBU of 0.3, compared to 0.5 for processor or memory utilization. Also, the RSE is 2.2 for a BBU of 0.8 compared to 1.5 when processor utilization is 0.8 (Figures 1, 3).[4]

Bisection Bandwidth and Software Quality, Complexity, and Reliability

The reliability of a system will depend on the quality of the software, indicated by the defect rate, and it is well known that defect rates are higher when software becomes more complex [11]. As with processing and memory constraints, when bandwidth is limited, the software becomes more complex, which results in higher defect rates.[5] Often, one is forced to decompose the problem by task as opposed to data domain. Task parallelism is more asynchronous than data parallelism, introducing a load balancing problem and making synchronization more difficult. Also, it is often necessary to write low-level communication protocols and reduce communication, both of which are extremely difficult. The code must be made more efficient, which often requires programming in lower-level languages such as Assembly, resulting in complex code that is hard to understand. This can have a significant impact on the test and integration phase of a system as errors are more difficult to detect when code is complex. In fact, many errors go undetected until the operation and maintenance phase.

Figure 3. *Relation between RSE and BBU.*



Bisection Bandwidth and Software Maintenance

Software maintenance, which includes fixing defects and upgrading functionality, is generally the most costly phase of the lifecycle [10]. For example, the cost to develop 236,000 lines of code for an F-16 fighter system was $85 million, whereas the maintenance cost was $250 million [3]. It is extremely difficult to fix defects and upgrade functionality on complex systems. When part of the system is changed, the effect on the rest of the system must be determined, which requires considerable testing to ensure that the system is fully operable. For high-reliability systems, 75 percent of the time in an upgrade cycle is spent in testing and analyses [10]. With a bandwidth-constrained system, the maintenance phase requires more effort as the added complexity makes it more difficult to test the system.[6]

Upgrading functionality can put increased demands on communications resources, which results in an increase in the utilization of available bisection bandwidth, thus making software development and testing increasingly difficult[7] (Figure 3). It may also be necessary to reallocate bandwidth on the original application to accommodate the added functionality, which requires additional software design and development. If bandwidth is constrained in the original application and upgrades continually require additional bisection bandwidth, software development will become increasingly more difficult until upgrades are no longer possible.

Advantages of Additional Communication Resources

Given the substantial impact of bandwidth constraints on software development and maintenance costs, it may be strategically advantageous to invest in high-bandwidth interconnection networks or to develop new interconnection technologies. One promising technology is free-space optics in which data is transmitted through free space by unguided optical beams. The Defense Advanced Research Projects Agency (DARPA) recently initiated the Free-Space Optical Interconnect Accelerator Program with the intent to transfer this technology to military systems.

Besides providing direct savings in software costs, additional advantages can be realized by increasing bandwidth. This includes greater capability in Department of Defense (DoD) radar and imaging applications by enabling communication-intensive algorithms and by effectively implementing shared-memory systems. Greater capability may have direct impact on mission effectiveness, i.e., reduced loss of aircraft and life, and, therefore, on national security.

It is widely recognized that shared-memory systems are easier to program than message-passing systems that require extensive “tuning” to achieve optimal performance through locality and are more difficult to modify [8]. A shared-memory machine, with uniform memory access, does not require the programmer to be concerned about data locality to achieve optimal performance, which makes it easier to develop, maintain, and reuse code. However, to effectively implement locality independence and make dynamic load balancing effective, a high bandwidth interconnection network is required [8].

Multiprocessor Model for Software and System Cost Optimization

A number of parametric models have been developed to estimate software development costs for uniprocessors. These models can be used to estimate software effort and related factors such as schedule length once one determines the size of the software and attributes inherent in the project and development process, including

- Product complexity and reliability requirements.
- Memory and processor constraints.
- The level of application and programming experience of the employees.
- The use of modern programming practices and development tools.

To apply these models to multiprocessor systems, the impact of the interconnection network is included implicitly through attributes such as product complexity. However, this does not enable one to exploit the full benefits of modeling, i.e., trade-off analyses, to optimize software and system costs. For meaningful optimization, the relations between individual multiprocessor hardware components (processors, memory, and interconnection network) and software productivity and quality must be explicitly included in the model. Additional factors to consider are the number of processors and the interdependencies between major hardware components, e.g., to accommodate for bandwidth constraints requires increasing processor and memory requirements.

Developers of real-time multiprocessor systems have expressed a desire for such a model. A model of this nature should be extremely useful for the rapid design and prototyping of cost-effective real-time multiprocessing systems. It would enable trade-off analyses to be made in the early stages of the development cycle, e.g., conceptualization, and would support decisions on high-level issues such as technology choices. It would also be desirable to include the maintenance phase in a real-time embedded software estimation model. This would allow one to optimize for upgrades and enable trade analyses based on the entire lifecycle of the system. We believe a modeling tool of this nature would produce substantial savings in costs over the lifecycle of an application.

Summary

Existing models, developed for uniprocessors, consider processing and memory constraints but do not consider parameters unique to multiprocessors such as bisection bandwidth constraints or number of processors. We have found that bisection bandwidth constraints impact software development as much as or more than processing and memory constraints. For this reason, the importance of bisection bandwidth should not be overlooked when estimating software costs for real-time embedded multiprocessors.

This work provides a basis to extend existing parametric software cost-estimating models to describe real-time embedded multiprocessor systems. Such a model would make it possible to perform trade analyses to optimize system cost and performance, which will lead to substantial savings in development and maintenance costs, increased performance, and easier upgrades. ◆

Acknowledgments

A number of people contributed to this work. Special thanks to Jim Otte, of PRICE Systems, for useful discussions on cost-estimation relationships in parametric models and software effort data. We also thank Barbara Yoon for providing information on application domains to pursue and people to contact. This work was supported by DARPA under contract F30602-96-2-0234, managed by Rome Laboratory, Optoelectronic Computing Systems Center.

About the Authors

Wes C. Berseth is a professional research associate of the Optoelectronic Computing Systems Center at the University of Colorado. His research includes the application of optoelectronics to imaging and communications systems from an architectural, performance, and cost perspective including the impact of hardware resources on software productivity and quality. He is currently principal investigator of modeling and simulation on a DARPA-sponsored Free-Space Optical Interconnect Accelerator Program. He has a bachelor's degree and a master's degree from York University, Toronto, Canada, and holds a doctorate in physics from York University. He is a member of the American Physical Society, Association for Computing Machinery, and the Institute of Electrical and Electronics Engineers (IEEE).

Optoelectronic Computing Systems Center
Campus Box 525
University of Colorado
Boulder, CO 80309-525
Voice: 303-492-0478
Fax: 303-492-3674
E-mail: wberseth@colorado.edu

John A. Neff is director of the Optoelectronic Computing Systems Center at the University of Colorado. Before joining the University of Colorado in 1991, he was the research manager for optical computing at DuPont. Prior to joining DuPont in 1988, he spent 15 years as a program manager for optical computing with the DoD, first with the Air Force Office of Scientific Research, then with DARPA. He has a bachelor's degree in physics and in mathematics from Ohio Wesleyan University and a master's degree and doctorate in electrical engineering from Ohio State University. He is a fellow and past governor of the Society of Photo-Optical Instrumentation Engineers and is a fellow of the Optical Society of America. He is listed in Strathmore's Who's Who of Business Leaders.

Director
Optoelectronic Computing Systems Center
University of Colorado
Campus Box 525
Boulder, CO 80309-0525
Voice: 303-492-7135
Fax: 303-492-3674
E-mail: jneff@colorado.edu

Anis Husain is assistant director of the Electronics Technology Office at DARPA. From March 1994 until May 1997 he served as program manager of optoelectronics in the DARPA Microelectronics Technology Office (MTO) re-

sponsible for research and development in optical interconnects, free space optical interconnects, visible emitters and detectors, and high-density optical memory. Prior to this, he was affiliated with the Center of High-Performance Computing, Worcester Polytechnic Institute, Boston, Mass. as an on-site consultant to DARPA/MTO. From 1980 to 1995, he worked for Honeywell Technology Center as supervisor of media and network architecture and section manager of photonics. He has a bachelor's degree in electrical engineering from Imperial College, University of London, United Kingdom and holds a doctorate from University College London, United Kingdom. He is a member of the IEEE, IEE, Optical Society of America, Computer Society, Communications Society, and Lasers and Electrooptics Society.

Optical Micro-Machines, Inc.
6160 Lusk Blvd., Suite C205
San Diego, CA 92121
Voice: 703-758-2622
Fax: 619-642-0490
E-mail: ahasain_omm@ibm.net

Bill Wren is a principal research scientist at the Honeywell Technology Center in Minneapolis, Minn. He has broad experience in scalable real-time processor architecture design, system modeling and trades, digital design, signal processing design and programming, software architecture design, and program management. He has a bachelor's degree in electrical engineering from the University of Nebraska. His graduate studies included work at Arizona State University, and he received a master's degree in electrical engineering from the University of St. Thomas, where he studied software systems.

Honeywell Technology Center
18820 DeAnn Circle
Minnetonka, MN 55345
Voice: 612-951-7885
E-mail: wren_bill@htc.honeywell.com

References

1. Donahue, Lt. Gen. William J., "Information Dominance Through Software Technology," *CROSSTALK*, STSC, Hill Air Force Base, Utah, July 1997, pp. 3-4.
2. Stutzke, Richard D., "Software Estimation Technology: A Survey," *CROSSTALK*, STSC, Hill Air Force Base, Utah, May 1996, pp. 17-22.
3. Bunza, Geoff J., "A Journey into Parallel Worlds: Exploring Hardware/Software Systems Integration," *Embedded Systems Conference*, San Jose, Calif., September 1996, Miller Freeman.
4. Bartow, James, "Evolutionary Design of Complex Systems," *Investments in Avionics and Missiles Software Technology Workshop Report, ARPA/SISTO*, Software Productivity Consortium, Inc. Report SPC-95068-CMC, 1995.
5. Minkiewicz, Arlene and Anthony DeMarco, "The PRICE Software Model," PRICE Systems, June 1996.
6. Anderson, James. C., "Projecting RASSP Benefits," *Proceedings of the 2nd Annual RASSP Conference*, ARPA, Arlington, Va., 1995, pp. 65-72.
7. *Accelerated Strategic Computing Initiative (ASCI) PathForward Project Description*, Dec. 27, 1996, http://www.llnl.gov/asci-pathforward.
8. Probst, D.K., "Architectural Visions Versus Business Models: How Soon Will There Be Enabling Technologies for Petaflops Computing?" Report prepared for Gil Weiland at DoE/DP-07 [HQ] December 1995.
9. Blelloch, Guy E., B.M. Maggs, and G.L. Mile, "The Hidden Cost of Low Bandwidth Communication," *Developing a Computer Science Agenda for High-Performance Computing*, ACM Press, 1994, pp. 22-25.
10. Shrobe, Howard, "Evolutionary Design of Complex Software," *Investments in Avionics and Missiles Software Technology Workshop Report, ARPA/SISTO*, Software Productivity Consortium, Inc. Report SPC-95068-CMC, 1995.
11. Jones, Capers, *Applied Software Measurement: Assuring Productivity and Quality*, 2d ed., McGraw-Hill, New York, 1997.

Notes

1. Bandwidth is defined as the rate at which an interconnection link can transfer information. For digital systems it is measured in bits per second.
2. Bisection bandwidth provides a measure of the communication resources of the network. For a symmetric network, bisection bandwidth is determined by dividing the interconnection network into two equal parts, each with half the processing nodes, and summing the bandwidth of all lines crossing the dividing line.
3. The corner-turn, also called a transpose, is an "all-to-all" communication operation in which the processors on the network send data to each other in preparation for the next computation operation. This operation is important in certain signal-processing applications, e.g., two-dimensional fast Fourier transform and SAR, and can severely overload the network and stall computation.
4. This was supported by the questionnaire as seven of 10 claimed that bisection bandwidth constraints can affect software development as much as or more than memory constraints, and eight of 10 claimed that software development is affected as much as or more than when processing is constrained.
5. Ten of 11 experts surveyed claimed that bandwidth constraints increase software complexity and six of six claimed that bandwidth constraints affect defect rate by making code more complex.
6. Eight of 10 experts interviewed stated that bandwidth constraints make the maintenance phase more difficult because the added complexity makes it more difficult to test the system.
7. Seven of 11 experts interviewed stated that adding new functionality to a previously developed application will use additional bandwidth. The remaining four said it can, depending on whether the tasks are scheduled concurrently.

Sponsoring Organization Model for Information Technology Certification Systems

William H. Dashiell, *National Imagery and Mapping Agency*
Arnold Johnson, *National Institute of Standards and Technology*
Phil Brashear, *Electronic Data Systems*

Current acquisition reform trends have challenged acquisition managers to acquire software on time and within budget. The National Institute of Standards and Technology (NIST) Sponsoring Organization Model (SOM) can help acquisition managers and acquisition organizations ensure that the software they acquire fits their needs and is of acceptable quality, which in the long term will save organizations significant time and money. This article sets forth the need for and benefits of an SOM, presents a basic overview of the SOM, and gives three brief example SOM implementations.

Many government information technology (IT) acquisition managers and program managers acquire computer software or contract for the acquisition of special-purpose software to be maintained by the organization acquiring the software. To meet procurement (program) requirements within cost constraints, acquisition managers and program managers need to use all cost-reduction means or resources available. Since conformance to recognized standards is one indication of the completeness and maintainability of software products, conforming software can help hold down costs while meeting program requirements.

Recent Acquisition Reform

In 1996, both the U.S. Congress and the administration initiated efforts to streamline the federal government's acquisition activities by reducing the central management structure and strengthening the authority of each agency's acquisition decisions. As part of this activity, Congress passed the Information Technology Management Reform Act of 1996 (Division E of Public Law 104-106), and President Clinton signed Executive Order 13011, which emphasizes agency management of information technology and new government-wide interagency support activities to improve productivity, security, interoperability, and coordination of government resources. Public Law 104-133 emphasizes federal government use of voluntary industry standards and directs federal agencies to use voluntary standards and to participate in their development.

There is no government-wide requirement for certification of acquired software that is meant to conform to an IT standard nor is there a general requirement that all acquired software be developed using certified tools. However, standardization has enormous implications for maintainability and portability for a program manager, especially since the bulk of software costs are incurred during maintenance (including the porting of software to evolving hardware and operating systems). Therefore, it is important that all acquisition efforts take into consideration the specification of IT standards and the use of development tools (compilers, in particular) that have been tested for conformance to standards.

Good software development practices require that software meet appropriate standards. Some of these standards are maintained by various entities, historically including the NIST, the American National Standards Institute, the Institute of Electrical and Electronic Engineers, and the International Organization for Standardization.

Conformance to software standards is normally established by means of a validation test suite. Based on the successful results of processing these test suites under third-party observation, software products are validated as conforming to the appropriate standard.

Each software purchaser or user must determine whether products with nonconformities are acceptable as meeting their needs. A software validation program may provide the information for a better software selection.

Standardization of testing methods and criteria for conformance to selected IT standards allows developers of IT software to verify conformance to those standards. Verifiable conformance is important to meet procurement requirements, to allow interoperability of various software products, and to allow for the use and sharing of data among various software products. Validation testing by an independent third party, using a standardized conformance test suite, provides the best assurance that the developer has made a significant effort to comply with the appropriate standard.[1]

The Basic Model

Before delving into specific uses and applications of the SOM, it is important to understand the fundamental structure of the model as defined by the NIST. Implementations of the SOM are conformance testing (certification) systems. We will refer to these implementations as SOM certification systems (SOMCSs). An SOMCS (which can be tailored to fit an organization's needs) consists of the

- Sponsoring organization – the organization responsible for the software certification system's management, processes, and funding.
- Test method executive control committee – a committee of testing and process experts taken from all parties involved in the sponsoring organization. They are responsible for establishing the accepted testing methods and the requirements for conformity to a specified standard.
- A certificate-issuing organization – the NIST SOM also allows for multiple certificate-issuing organizations, each sponsoring one or more testing laboratories. This gives the sponsoring organization testing coverage for a wide variety of software types.
- One or more testing laboratories – testing organizations recognized by the certificate-issuing organization as being qualified to test specific types of software against specified standards.

The basic (tailorable) NIST SOM for an IT certification system consists of two parts: a list of suggested functions (Table 1) and a list of suggested roles (Table 2). The contents of both lists were derived from NIST's test development and validation experiences. These lists are not all inclusive. An acquisition organization may require functions and roles not included in the NIST lists or may not require some of the functions or roles. To implement the model, a sponsoring organization chooses required functions and roles, maps the functions to the roles, and assigns specific organizations to the roles.

The central element in the resulting certification system is the sponsoring organization, which plays a key role in establishing and maintaining the system. The sponsoring organization may be any authority that assumes responsibility for the certification system. It may be composed of any combination of the following organizations: consortia, government agencies, or private software industry. Together, they work to broaden the scope of certification recognition.

Use and Benefits of the SOM

How can acquisition managers or program managers take advantage of existing SOMCSs (such as those sponsored by the Air Transport Association, the U.S. Geological Survey, or the Electronic Data Systems (EDS) Conformance Testing Center [explained later in the article]) to make the best software selection? Neither acquisition managers nor program managers use the SOMCS directly but depend on products and results from the SOMCS. For example, acquisition managers can develop better solicitation requirements by requiring the use of products with certificates from existing SOMCSs or can develop their own SOMCS for this purpose. The program manager can use the information provided by an SOMCS to select a tested product that best meets the program's needs, which saves the program money. This "get-it-right-the-first-time" method results in lower software lifecycle costs compared to low-cost software that may not meet all the chosen standard's requirements and therefore may require an inordinate amount of maintenance and modification.

The SOM is applicable to acquisitions or programs in which

- A software product is a deliverable, and a recognized standard exists to which the product should conform, such as C++ compilers, which should conform to the (not yet completed) International Standard or mapping information databases, which should conform to the Topological Vector Profile of the Spatial Data Transfer Standard.
- A software product is to be used in the development of a deliverable, and the government must maintain the deliverable, such as delivery of a command and control software system to be developed using standardized programming languages or delivery of a design specification in which all schematics are to be delivered in a format conforming to the Computer-Aided Acquisition and Logistics Support profile of the Computer Graphics Metafile (CGM) standard.

In these cases, the solicitation should specify that all such software products be tested by a certification system based on the SOM. If the acquiring agency is a sole or joint sponsor of an SOMCS, the solicitation can specify their SOMCS as the source of certificates.

The benefits of using an SOMCS include

- (First case, above) the assurance that delivered software products have been subjected to standardized testing procedures, using known test instruments, under the supervision of a disinterested third party. This provides acquisition managers with a set of objective assessments on which to base a procurement decision.
- (First case, above) some assurance that government use of delivered software products will have predictable results (as promised by the relevant standard).
- (First case, above) assurance that government employees who use delivered software products will not have to be retrained to use non-standard features.
- (Second case, above) assurance that deliverables developed using the

Table 1. *Certification system functions.*

Establish policies and procedures.
Recognize certificate-issuing organizations.
Resolve technical and procedural issues.
Approve content and use of the test suite.
Issue validation certificates.
Maintain a public list of validated products.
Recognize testing laboratories.
Maintain conformance test suite.
Conduct conformance testing.

Table 2. *Certification system roles.*

IT Standard Committee
Sponsoring Organization
Test Suite Developer/Maintainer
Advisory/Control Board
Certificate-Issuing Organization
Technical Reviewers/Experts
Testing Laboratory
Users of SOM Products
Validation Customer

tested software products can be modified as needed, using either the tested software products or other products known to conform to the same standard.
- (Second case, above) assurance that deliverables have known interface characteristics, e.g., data sharing, and that software using those interfaces will have specified behaviors as specified by the relevant standard.

Solicitation Wording

Because of Public Law 104-133, software procurement solicitations should contain requirements for conformance testing of IT products when the delivered products are expected to conform to an IT standard or when they are used to develop products to be delivered to and maintained by the acquiring organization. Major benefits are the existence of a software standard against which to compare when the software does not function properly and the increased assurance that the software will be maintained by the vendors and that interfaces with other conforming software will function as expected by the standard. Specific procurement activities have the authority to determine the demonstrated degree of conformance (either zero nonconformities or some limited number of nonconformities exposed by testing).

But why would one knowingly purchase software with known nonconformities? The NIST SOM requires that a Validation Summary Report be written for each validation effort. The Validation Summary Report is produced by a testing laboratory and contains the results that are observed from validation testing of a specific software product under test. Acquisition managers and program managers who review the Validation Summary Report for software that fits their requirements may find that the nonconformities exist in functional areas that are irrelevant to their program's needs. For example, they may be seeking a standardized application language compiler for an embedded system that has no need for text input and output. In this case, a nonconforming compiler that does not support text input and output may provide the best value—a potential for significant savings.

The NIST has developed suggested wording for acquisitions for which zero nonconformities are allowed or acquisitions for which limited nonconformities are allowed. (http://sdct-sunsrv1.ncsl.nist.gov/~ftp/vpl/validwrd.htm). Those acquisition managers and program managers whose needs require full conformance should select the zero nonconformities wording.

In addition to allowing variations in the number of nonconformities, the suggested solicitation wording allows acquisition managers and program managers to select one of three validation options: *delayed validation, prior validation,* and *prior validation testing (with errors)*.

Delayed validation allows IT suppliers to offer products that may not have been tested prior to contract award but must be tested during the contract period. This option would allow an acquisition manager to purchase a software product being developed (perhaps state of the art) and know the product will be assessed for conformance to the standard during the contract period. The risk is that the software may not meet the requirements of the standard, and, therefore, the procurement manager may negotiate a better price.

Prior validation requires product suppliers to have their products validated with zero nonconformities prior to contract award. Here, the acquisition manager and the program manager have the greatest assurance that the product meets the standard.

Prior validation testing (with errors) requires that the products be tested prior to being offered in response to the solicitation request and allows for testing results exhibiting nonconformities. Those exhibited nonconformities, summarized in a Validation Summary Report, may not be important to the program needs or software users and may represent a cost savings to the acquisition manager and the program manager.

Extrapolation from Validation Results

With ever-shrinking budgets, it is not feasible to directly test all candidate IT products for conformance, since formal validation can be expensive, time-consuming, and resource-intensive. Acquisition managers may decide to extrapolate information from the test results published by the certificate-issuing organization based on additional research, demonstrations, or warranties by the IT product supplier. It must be kept in mind that a validation certificate attests only to the successful testing of a product in a particular environment (hardware and system software). One cannot assume that the conformance of a product in a particular environment implies the conformance of a different version of the product (even from the same implementer) or the same version in a different operating environment. It is the acquisition manager and the program manager's responsibility, not an outside organization's, to review the certificate-issuing organization's Validated Product List and determine the applicability of these validations to the needs of the acquisition manager and the program manager. The applicability and usefulness of a validation certificate should be based on the size and timing of the procurement.

SOM Example Implementations

The NIST Directory of Conformance Testing Programs, Products and Services (http://www.itl.nist.gov/div897/ctg/ctdhome.htm) lists some existing testing services, including the following current implementations of the NIST SOM. The descriptions below illustrate some of the variations possible in implementing the model.

Government Agency with Commercial Certificate-Issuing Organization

The U.S. Geological Survey (USGS), an agency of the Department of the Interior, sponsors a certification system (Figure 1) for IT products that implement the Spatial Data Transfer Standard, Topological Vector Profile (SDTS TVP).

This standard specifies formats for the transfer of spatial data among different computer systems. The USGS, with assistance from the NIST, has developed a test suite to validate products that implement SDTS.

The USGS (http://mcmcweb.er.usgs.gov/sdts/conform.html) has recognized the Conformance Testing Center (CTC) at EDS (http://eds-conform.com/SDTS.html) as both a certificate-issuing organization and a testing laboratory for the SDTS TVP certification system.

Trade Association with Commercial Testing Laboratory

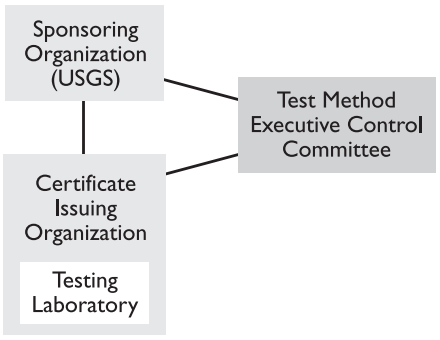
The Air Transport Association (ATA), a trade association, has established a certification system (Figure 2) for products that implement the CGM ATA Profile (as defined in ATA 2100 Specification, Graphics Exchange).

The NIST CGM Interpreter Test Suite is used to validate interpreters for the CGM ATA. The NIST CGM ATA interpreter test service is expected to be terminated in 1998, when the ATA establishes a CGM testing program. The ATA, serving as both sponsoring organization and certificate-issuing organization for its certification system, has solicited for testing laboratories (http://www.itl.nist.gov/div897/ctg/graphics/cgm.htm).

Commercial Sponsoring Organization

In reaction to the withdrawal of the NIST and the Ada Joint Program Office (AJPO) from validation of language processors (compilers), EDS operates a compiler certification system (Figure 3)

Figure 1. *SDTS TVP certification system.*



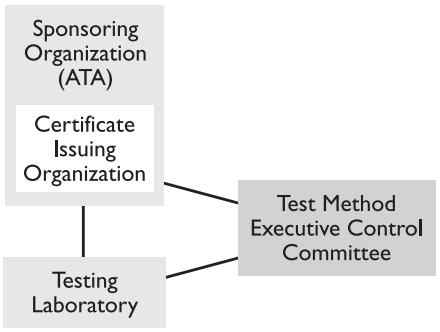


Figure 2. *ATA CGM certification system.*

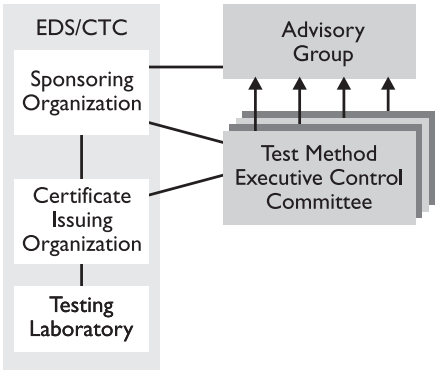


Figure 3: *EDS CTC compiler certification system.*

as an implementation of the NIST SOM. This system provides validation testing services and issues certificates for Ada 83 and 87, Ada 95, C, COBOL 85, and FORTRAN 77 compilers as well as Structured Query Language processors. In most cases, the test suites used for these validations are the ones that the NIST and AJPO have used in the past. Plans to test C++ compilers are underway (http://eds-conform.com).

As Figure 3 shows, EDS CTC fulfills all three major roles: sponsoring organization, certificate-issuing organization, and testing laboratory. A test method executive control committee is established for each standard, with the majority of the members drawn from the validation customers and organizations implementing the standard. Each test method executive control committee advises EDS CTC on policy and procedures specific to validations for its standard as well as controlling the test suite and resolving validation issues related to that standard. An advisory group provides advice on the overall certification system policy and procedures. Each test method executive control committee names one of its validation customer members to serve on the advisory group.

The EDS CTC certification system has only one testing laboratory, which is operated by the CTC. However, procedures are in place to recognize other testing laboratories and to issue certificates in accordance with their recommendations.

Conclusion

Government acquisition reform emphasizes decentralized control of procurement. At the same time, there is a strong trend to privatize functions previously performed by the government. These forces increase both the power and the responsibility of the acquisition manager and the program manager. In particular, these managers are empowered to take advantage of SOMCSs to ensure the acquisition and use of software that conforms to standards. Doing so can significantly reduce development and maintenance costs while resulting in more reliable systems with predictable behavior. ◆

About the Authors

William H. Dashiell is a computer scientist at the Department of Defense National Imagery and Mapping Agency. He has worked on the development of software testing by statistical methods using binomial models, coverage designs, mutation testing, and usage models. He has contributed to the development of conformance and testing protocols for federal, national, and international information technology standards. He has a bachelor's degree in business administration and education, a master's degree in education technology, and a doctorate in mathematics education from the University of Maryland. He also has a master's degree in computer science from Hood College in Maryland.

National Imagery and Mapping Agency
Washington Navy Yard, Building 213

SNIT Mail Stop N-61
1200 First Street, S.E.
Washington, D.C. 20303-0001
Voice: 202-863-3516

Arnold Johnson has spent more than 20 years managing the development and operation of testing and certification programs spanning 10 federal and international IT standards including negotiating six bilateral or multilateral international agreements for mutual recognition of test results. He is currently the NIST program manager for the National Information Assurance Partnership, a joint initiative by the National Security Agency and NIST to foster the development of tests, test methods, tools, and commercial testing laboratories to evaluate the quality of IT security products.

National Institute of Standards and Technology
Information Technology Laboratory
Building 820, Room 517
Gaithersburg, MD 20899
Voice: 301-975-3247
E-mail: arnold.johnson@nist.gov

Phil Brashear is a senior systems engineer at EDS in Dayton, Ohio, where he leads the EDS Conformance Testing Center and the maintenance of the Ada compiler validation test suite. He previously directed compiler validation efforts at CTA Incorporated from 1989 to 1997. From 1986 to 1989, he performed compiler validations and directed test suite enhancements at SofTech, Inc. Prior to 1986, he was a member of the mathematics and computer science faculties at Eastern Kentucky University. He has a bachelor's degree in mathematics education from the University of Kentucky, a master's degree in mathematics from Northwestern University, and has completed the course work and examinations for a doctorate in mathematics at the University of Georgia.

Phil Brashear
EDS Conformance Testing Center
Dayton, Ohio
E-mail: phil.brashear@acm.org

Additional Reading

1. ISO/IEC Guide 2, *General Terms and Their Definitions Concerning Standardization and Related Activities.*
2. ISO/IEC Guide 25, *General Requirements for the Competence of Calibration and Testing Laboratories,* 3rd ed., 1990.
3. ISO/IEC Guide 28, *General Rules for a Model Third-Party Certification System for Products*.
4. ISO/IEC Guide 38, *General Requirements for the Acceptance of Testing Laboratories.*
5. ISO/IEC Guide 39, *General Requirements for the Acceptance of Inspection Bodies.*
6. ISO/IEC Guide 40, *General Requirements for the Acceptance of Certification Bodies*, currently under revision, April 1995.
7. ISO/IEC Guide 42, *Guidelines for a Step by Step Approach to an International Certification System.*
8. ISO/IEC Guide 43, *Development and Operation of Laboratory Proficiency Testing.*
9. ISO/IEC Guide 44, *General Rules for ISO/IEC and IEC International Third-Party Certification Schemes for Products.*
10. ISO/IEC Guide 56, *An Approach to the Review by a Certification Body of Its Own Internal Quality System.*
11. ISO/IEC Guide 60, *Code of Good Practice for Conformity Assessment*.
12. ISO/IEC Technical Report 13233, "IT – Interpretation of Accreditation Requirements," ISO/IEC Guide 25, *Accreditation of IT and Telecommunications Testing Laboratories for Software and Protocol Testing Services*, Nov. 30, 1995.
13. Rada, Roy, "Who Will Test Conformance?" *ACM Communications*, January 1996.
14. *Test Method Control Procedures Model, Workshop on Harmonization of Programming Languages and Graphics Validations*, March 15-16, 1994, NIST, Computer Systems Laboratory.

Note

1. Two concepts must be emphasized. First, validation (conformance) testing does not warrant that the product tested is free of nonconformities, even if all tests are passed. Second, validation testing is not intended as a means of performance benchmarking.

WebTALK: A New On-Line Discussion Forum

CROSSTALK is based on the premise that sharing information is the fastest way to learn. The software engineering field, still in its infancy, is still trying to define itself–no other industry can serve as a model for the process and techniques needed to produce good software. Trial and error is still the predominant, but we hope doomed, method.

With that in mind, we hope you will share your ideas about software development–or your reaction to ideas presented in *CROSSTALK*–via a new feature on our Web site: WebTALK, an on-line forum that affords you the opportunity to engage in some cross talk of your own. The discussions are formatted as threads to make conversation as easy as a mouse click. Access WebTALK from *CROSSTALK*'s home page (http://www.stsc.hill.af.mil/CrossTalk/crostalk.html).

Your ideas count. Be heard!

Searching for Good Help?

Mike Duffy
Tec-Masters, Inc.

On-line help authoring tools are the genies that can help you make the transition from hard copy to screen documentation. This article describes the lessons learned from a search for effective and efficient on-line help authoring software. It is a look "behind the screen" to describe what to look for in an on-line help writing package and also how help can enhance and modernize documentation.

HELP. HAVE YOU EVER USED SUPPLEMENTARY application software that made you want to pull out your hair? Did it make you want to use a few expletives or question your abilities? This article records knowledge and experience gleaned from using one such conglomeration of programmed overlays, paths, and interfaces. This article is not a mystery but a look at on-line help authoring applications from a writer or a user's perspective rather than from a technical evaluator's.

On-Line Help Defined

If you are looking for an on-line help authoring package or have never used one, these experiences may prove helpful. Most help tools use or overlay Microsoft Word™ and Microsoft On-Line Help™ to make it easy to write your own on-line help for your project or system. Writing on-line reports and documentation with embedded jumps, indexing, pop-ups, definitions, cross-referencing, and navigation can be a relatively easy task with the proper tool. The term "on-line" here refers to a report or document displayed on the screen after the inclusion of the above features. Such a product would be on line in the conventional sense only when it was connected to a local-area network, wide-area network, Intranet, or Internet.

Two sample screens will demonstrate how on-line help works. Figure 1 shows the table of contents for a section in a user's manual. The user selects a subject and the program goes to the specific paragraph. Note that no help programming characters are shown on this screen but are in Figure 2, "Civil Engineer Material System (CEMAS) Control File," a sample jump. In these examples, the traditional decimal paragraph numbers were retained, and the text was condensed. You may be able to further improve your documents.

Modernizing Documentation

Using a help authoring package in the documentation arena can enable you to modernize and condense the required "standardese" content of document formats. You can compose your original with the help authoring application. It will allow you to write the required paragraphs with the facts, guidelines, and instructions in easy-to-read, well-spaced lists or action subparagraphs with little running text. You can jump to a definition, figure, or an attachment for more information to facilitate brevity in your text. An on-line help authoring tool set can also be used to improve and enhance your documentation with pop-up boxes and cross-referencing.

Empowering the User

You may also transport and edit down an existing draft. At the porting point, if you do not modernize the style of your document, you can almost lose the reason for on-line documentation. Remember to empower your users—with an on-line help authoring tool, standards can be met in plain screen English. The user can move from general to specific and back without turning pages, and reading time can be cut in half. You can go to a definition and back again and then to related information with ease, accuracy, and speed. The documentation can more

Figure 1. *Screen 1. Table of contents in user's manual.*

```
SECTION 6. CEMAS FILE MAINTENANCE
   CEMAS System Access Table of Contents:
6.1      CEMAS System Access
6.1.1       CEMAS Identification File (MIDF)
6.1.1.1        Add/Inquire-ID Number Program (MIDFUD)
6.1.1.2        Identification Number Update Screen
6.1.1.3        Identification Number Add Screen
6.1.2       CEMAS Processing File (MPRO)
6.1.3       CEMAS Control File (MCTL)
6.1.4       CEMAS Program Access
6.1.4.1        CEMAS Master Menu Program (MCEMMENU)
6.1.4.2        CEMAS System Administrator Program (MCEMSSA)
```

Figure 2. *Screen 2. Paragraph from the selection in Screen 1. Note: The characters "#$+K" are software-generated codes that establish the location and indexing of this paragraph. The paragraph number and title are automatically bolded and the text appears as shown here. In a finished help file, you would not see the coding shown here for illustration.*

```
#$+K 6.1.3 CEMAS CONTROL FILE (MCTL).

The MCTL file is used to store control information for
CEMAS operations. This record contains such information as

  • End-of-Day (EOD) flag to indicate if EOS is processing.
  • Document Serial Numbers to be used in assigning document
    numbers to transactions.
  • CEMAS Stock List (CSL) number to be assigned to the next
    record added to the Noun Dictionary file.
  • Printer numbers used in printing products.

The information in the file is displayed on the CEMAS Base
Variable screens.
```

closely match the system and its inherent features when you apply some editing skill with a good on-line help authoring package. Remember, when you write you are building in the navigation, which becomes as easy as a mouse click for the user.

The First Attempt

When we started our relational database project with a major database management system, it appeared that we would have a computer-assisted software engineering tool to help us generate the documentation. The system did produce context-sensitive information from the hierarchical and attribute realms but stopped there. This led to searches for on-line help authoring tools to create documentation outside the main tool suite.

After some inquiries, we found a small help authoring application and started our project. This package, however, did not live up to its billing or our expectations. At times, when using our first help tool set, some of the commands that called in a template to set up a file did not work. Sometimes the indexing would have errors that were difficult to correct if a particular file in a series had an error. This error, in turn, would cause problems later in navigating from a subject to related material or to another subject. Setup commands and sometimes table of contents commands had to be repeated, which was both time-consuming and frustrating. This had a multiplier effect in relation to time and effort; however, we did complete a few small help files.

The Second Attempt

After pushing our first small system and ourselves almost to the limits with the first product, we searched the Web for another package. We found one with a good tutorial that taught from initial setup to full creation of a dummy help file. It included jumps, hot spots, indexing, and generally explained the whole package along the way.

Finding a help tool with a good range of capabilities proved highly beneficial. We had found an on-line help authoring package that was easy to use, accurate, and efficient in doing its job. The interactive tutoring enabled the creation of a dummy file and also the changing of an actual file into one with on-line help. It also included the ability to import figures. The limitation in the demonstration copy was that it only allowed 10 topics. Nevertheless, using the application helped us arrive at an objective decision whether to use it. We could also use a real file for practice with the tutorial—a real plus. As a result, we were able to go from package evaluation to project completion with less work and more accuracy; at this point we felt we had a good on-line help authoring tool set.

Lessons Learned

Our two experiences with on-line help authoring tools taught us the following principles.

Know What You Need

Take a close look at the papers or documents you are going to write. What features do you wish to include? Examine some text with on-line help included; this will give you an idea of what is available. To avoid the snags and pitfalls we encountered, thoroughly investigate the capabilities and shortcomings of any "help" for on-line help authoring package before you purchase it.

Get a Demonstration Copy

Get a demonstration copy if you can; some companies have "demo" copies available for download from the Web. This can save you both headaches and money, and it is also a good way to evaluate a software package before you buy it.

Make an Informed Decision

When you find a good package, complete the tutorial, then ask two or more co-workers to use it and give you their assessments. If you are not able to download a demonstration copy, study all the advertisements you can find on a product and call, write, or E-mail for more information. The bottom line is to make an informed decision. A decision made on one lone recommendation could cause a lot of aggravation.

Shop Around

If you decide to do an on-line help authoring document, first analyze what you want to do. Do you need jumps, pop-up boxes, indexing, attachments, table of contents, etc.? After a thorough analysis, start looking for an on-line help tool set that has the capabilities you want and that also has a good tutorial. Do your homework first, then start your search. Computer magazines and catalogs have hundreds of advertisements. We found the general price range was from $350 to $750 depending on the desired features and the complexities of the project. ◆

About the Author

Mike Duffy is a technical writer at Tec-Masters, Inc., (TMI). He serves as a documentation consultant and writer for TMI, a software development contractor in Montgomery, Ala. His previous positions include editor and technical manuals editor in the Quality Assurance Division of the Standard Systems Group at Maxwell Air Force Base, Gunter Annex before retiring with 33 years as an Air Force civilian. He has over 15 years experience as an editor in the automated information systems documentation field where he helped ensure Department of Defense and Air Force standards and editorial quality in base-level and command systems documents from the days of the Burroughs B3500 and Honeywell H6000 through the Unisys U2200. Currently, he writes and edits systems documentation.

Tec-Masters, Inc.
2430 Fairlane Drive, Suite C-7
Montgomery, AL 36116
Voice: 334-213-5438
Fax: 334-279-5286
E-mail: mduffy@mtg.tecmasters.com

If Nobody Uses It, It "Ain't" a Standard

Thoughts on Retooling DoD Data Standardization Efforts

Gary A. Ham and Douglas D. Mann
Battelle Memorial Institute

The Department of Defense should abandon the unified data model and single data representation approach to data standardization in favor of a shareable and open repository approach in which standards are chosen on the basis of quality and competitive merit.

Department of Defense (DoD) data standardization policy[1] (particularly as implemented in the DoD Data Model) has inherited much of its structure from the Corporate Information Management (CIM) concept. The data side of CIM, as implemented in DoD Directive 8320.1 [1], is based on the assumption that a single data structure, designed from the top down by selected subject matter experts, can be crafted to meet the needs of all development efforts.[2] True believers in CIM consider the admitted high cost of maintaining a single complicated relationship structure and a single approved representation for each information concept to be warranted in light of the benefits to be received. Benefits cited include well-defined, usable data structures, effective reuse of data in multiple systems, and higher-quality systems with lower maintenance costs.

Standardization Problems

We are not true believers. There are at least three significant problems.

Independent Definition

First, usable data structures cannot be defined independently from system requirements. While the same data structures can and should be reused in multiple systems, their structure must first and most important be based on mission activities. Data does not exist independently of mission, and mission implies a functional requirement. The problem: Data standards "defined" without a direct tie to a specific mission requirement have no basis for standardization. They may exist, but they have no purpose.

Differing Requirements

Second, different systems have different missions and, therefore, different requirements. Although there may be common data structures in different systems, the relationships these data structures have with other data structures may be different. Trying to maintain them all in a single, over-arching model is complicated. It is often argued that data is easier to model than process because data structures are more stable than processes. Relationships, however, often represent the processes that connect data structures. Imposing the relationships defined in a single, highly detailed model inhibits appropriate reuse of the data structures in the model. The problem: Standard data models that impose fixed process-oriented relationships restrict process change just as rigidly as any hierarchically defined process model.

Standardization May Not Reduce Costs

Third, standardization does not necessarily improve software maintainability or save on maintenance cost. Standardizing internal data structures removes the benefit of module encapsulation because it creates unwarranted coupling[3] between systems. When a data structure must be changed for one system, it has a ripple effect on all other systems using the data structure. The net effect is the creation of brittle systems that cannot be changed effectively for fear of side effects. The larger and more comprehensive the "standard" data structure, the more pervasive this "quality killer" becomes. The alternative is to develop work-arounds to avoid changing the standard data structures. Such work-arounds impose increasing degrees of maintenance brittleness onto a system, which increases future costs and decreases the flexibility to introduce additional change. Perhaps the best example of data coupling in the real world is the Year 2000 problem. Fixing this one badly chosen standard[4] will be expensive. Imagine if the structure were a bit larger. The problem: Building multiple systems around a single standard data structure is likely to *add* cost and increase maintenance effort.

The Most Beneficial Standards

On the other hand, communication of any kind is impossible without standards. Neither humans nor systems can understand each other without understanding both representation (the commonly agreed-on "sign," such as a word, character, or gesture) and concept (the object or idea to which the sign points). The more widely used a language is, the more useful it is for general communication, regardless of the quality of language construction.[5] The writing system for the English

The opinions in this article do not necessarily represent those of Battelle or its DoD clientele. Data standardization in DoD has been difficult—the solution is open to debate. Critical response is welcome.

language, for instance, is a hodgepodge of conventions from several languages. Consistency is not its strong suit. Nevertheless, in a world where English is increasingly becoming the common language of the business world, poorly spelled English functions better than no common language at all. Data standards work the same way. The most beneficial standards may or may not be the best in terms of any arbitrary standard of quality; rather, they are the ones *perceived by the user* as the most beneficial, because of adoption and common usage.

Sometimes, standards can be imposed by a common authority. The Health Care Financing Administration (HCFA), for instance, will probably have some success with the individual standards it chooses to impose because it has the power of enforcement under the Health Insurance Portability and Accountability Act of 1996 [2]. Even HCFA will not succeed, however, if it chooses to impose standards that are not *perceived by the user or the developer* as usable. Simply put, no developer will attempt to achieve something that is not perceived to be possible. The complicated nature of the current DoD Data Model is not generally perceived by developers to be implementable. In fact, the Government Accounting Office (GAO) found that only nine of 43 major DoD systems had plans to use standard data [3]. Smaller systems, with lesser resource allocation, are probably even less compliant. It is just too hard.

DoD is not alone in the practice of building data models that are little more than shelfware. Developed primarily by IBM as a standards proposal, the *Information Resource Dictionary System-Information Model* (draft dated April 8, 1992) consists of 763 pages [4]. The model was developed as the IBM Information Model in MVS-based Repository Manager. It was probably an extremely expensive development project. Unfortunately, the model is so complicated that it was not adopted. Whether IBM has made other use of this document is unknown. There are undoubtedly many other examples (usually unpublished).

DoD Standards

If DoD (or any organization) wants a successful data standardization program, standardization authorities must recognize that they have two objectives: develop or adopt *usable* standards and convince users and developers that the standards are usable. If the first objective is not realized, the second will not be, either. Without the second objective, the first is useless.

Usable Standards

Usable standards development requires the participation of developers. It must be system requirements based. Data structures must track directly to defined system information requirements. Simply, if you cannot state a specific use for a piece of information, you cannot consider it usable for standardization. Development of usable standards means cooperation and teamwork with actual development systems. If no developers are actually using a standard you develop, it ain't[6] a standard.

Adoption of usable data standards implies that the standards are *already in use* somewhere. They may be industry, government, or standards-organization sponsored. Usability is a function of quality, but the real measure of usability is widespread acceptance and implementation. Standardization may require compromise where the most widespread standard is "not as good" as its less widely used competitor or the one developed in-house. The point is, there may be no one standard for any particular concept or representation of that concept. Instead, there may be several. The best standardization programs choose the "best" standards by reviewing them all against mission activity and system development requirements. It is conceivable that more than one representation of the same concept could be adopted to meet differing mission requirements.

Not a Top-Down Process

While requirements definition should be done from the top down to ensure completeness, effective use of data standards is not a top-down process.[7] Choosing or building standard components to meet functional requirements should be done at the level at which the requirement is to be implemented. Standards should apply only to that information that is brought in or sent out from the requirement. Data internal to a particular requirement solution should remain decoupled from its interface with other requirements. Externally visible data should be standard within its sphere of visibility, which means that a particular concept must use the same name and structure within its context of visibility. Each layer of encapsulated visibility must meet its own set of standards for that layer. If passed beyond that layer, data must be wrapped to the set of standards applicable the next layer of visibility. By "wrappered" encapsulation, internal changes to system structures are not held hostage to changes in outside standards. Similarly, changes needed internally within a system are less likely to cause external system side effects. Only the interfaces need to be maintained.

It is at the interface level that standardization is particularly important. Systems that must interface with another system's nonstandards-based interface or with several different sets of standards must maintain multiple interfaces—one for each standard and one for each nonstandard system. Choosing a particular set of standards at each context level, i.e., level of visibility, reduces this interface to one. On the other hand, if reducing the set of standards to one creates a highly complicated set of intricate relationships, the one level may be harder to maintain than a multiple set of interfaces. The trade-off must be managed.

Implementation Management

The management of standards implementation is a necessary but inherently difficult process. To be successful, standards must be used to interface between systems and system components at the same level of visibility, without inhibiting encapsulation at layers above and below that level. A particular data standard can be adopted for use at all levels, if warranted, but only at the interface definition should such an adoption be

enforced. In fact, effective encapsulation requires some separation between interfaces and internals so that changes to one do not require extensive changes to the other. Making everything the same may make it easier to write the initial code. Maintenance costs, however, can be expected to increase.

The best overall standardization guideline is to adopt the most widely used standard for interfaces in general. Implementation, however, should only be enforced at given levels of visibility. Standardization becomes the process of choosing the standard representation for data to be absorbed or provided at a given level of visibility. Standardization within a particular system should be left to that system. Data passed from system to system for systems managed or owned by a particular functional area should be standard in name and representation throughout the functional area. Data transfers between DoD systems should meet DoD standards. Data passed to or from commercial sources should meet the appropriate commercial standard even if an additional interface is required.

Standards Composition

Just as the level at which a standard is appropriate varies in scale, so does the composition of the standard. An adoptable standard may be as simple as an individual code list or the structure of a single data element. It may also be as complex as an entire system interface (or a defined interface to a commercial-off-the-shelf package). In the object-oriented view, adoptable standards will consist of interface definitions for reusable components, varying in size from a single object class to an entire system.

Standards Adoption

Standards adoption is a process, not a localized, one-time event. It means comparing requirements with existing standards, picking an appropriate one where available, adapting one where it "almost" meets needs, or developing a new one where requirements are not compatible with what is available. Success in such a process has nothing to do with "correct" model building. Success comes from adopting standards that can and will be used. The key to that success is access to competing standards and visibility of how they are used. In the marketplace of ideas, the most usable standards will be adopted. Poor definition and incoherent design will be abandoned. In some cases, the best design may not win due to early adoption and wide dissemination of an otherwise competent predecessor. The value of reuse may outweigh the quality of later improvements. This is a decision that must be based on functional requirements and available resources.

Standards Registry

In the marketplace world, there is no "standard" set of standards. There are, however, multiple standard-setting organizations that offer their goods to the world. A standards registry can be used as a tool to provide an effective marketplace for these standards. Standard-setting organizations act as registration authorities, entering their adopted standards to their own space on the registry. Other organizations can then adopt standards from the registry for their own use or put up competing standards of their own in their own space. An international standard, ISO/IEC 11179, Information Technology – Specification and Standardization of Data Elements [5], provides the foundation for defining a registry for data elements and concepts. The six-part standard addresses

- Framework for the specification and standardization of data elements.
- Classification for data elements.
- Basic attributes of data elements.
- Rules and guidelines for the formulation of data definitions.
- Naming and identification principles for data elements.
- Registration of data elements.

The draft American National Standard, dpANS X3.285, Metamodel for the Management of Sharable Data [6], takes the international standard and extends it into data-value domains and concepts. A draft technical report, Concept of Operations for a Data Element Registry of July 1996 [7], from the American National Standards Institute (ANSI) National Committee for Information Technology Standards L8 – Data Representation subcommittee addresses the operation of a registry based upon the International Organization for Standardization (ISO) and ANSI standards. A registry based on these standards has the capability of registering concepts, data elements, data-value domains, classification schemes, structures, and name contexts.

An organizational registry should support access to multiple registries for various registration authorities. When components may be viewed, compared, evaluated, and selected from multiple sources, the marketplace factors of quality and cost become important selection factors. Multiple registries also facilitate harmonization through cooperative consensus and peer pressure. Information sources and subject experts are identified for consultation. Registries also identify work in progress, approved future components, and older versions of components. An interesting side to the registry is that it improves data-standard quality by exposure to the public. What is not understood or is incomplete can be questioned and clarified.

Several organizations are developing registries based upon ISO and ANSI standards. The U.S. Environmental Protection Agency has the Environmental Data Registry. The Australian Institute of Health and Welfare has developed a health-care-related registry. The U.S. Census Bureau is about to release its registry. All these meet the international standard for registries. The Defense Data Dictionary System (in contrast to the DoD Data Model on which it is supposed to be based) also serves as a registry, although it does not provide the full functionality of the international registry standard.

Conceptually, the DoD Data Dictionary should be retooled to conform to international repository standards. Its management should act as the registration authority for information standards that apply at the DoD level. It should adopt standards from other registries where appropriate and make all such registries visible. It should pro-

vide a central registry and appropriate function area-based subregistries, overseeing a consensus-building effort toward mutually compatible systems interfaces based on well-defined, usable standards. Selected standards (particularly, systems interfaces and code domains used in multiple systems) could be dictated for use in all systems. DoD's current "new idea" in data standardization, the Defense Information Systems Agency-sponsored Shared Data Environment (SHADE) segment registration process [8], is an excellent beginning for this kind of well-grounded standards development process, although visibility to alternate registries would enhance quality and usability. In the meantime, attempts to require "standardization" for entire database structures internal to developing systems should be abandoned. Requiring such structures is actually quality inhibiting and not enforceable in any real sense.

Modeling

Modeling plays a different but extremely important role in this type of standards registration process. Instead of making sure that a potential standard meets the structure of some formal data model, standards that are proven to meet defined functional requirements are modeled to show their relationship with other adopted standards to improve their accessibility and provide opportunities for reuse. The change in focus is important. *Do not standardize the models. Instead, model the standards.* In this environment, functional area models are important navigational tools for using and integrating standards during systems development. Data relationships are modeled with data models. Component relationships are modeled as object models. Finally, all components must be mapped to a mission model.

Mission-based requirements models should be the *only* top-down-defined models in the DoD information management program. Even these models should be based on the required sets of measurable results needed to accomplish a mission rather than process steps involved in getting there. System functional requirements should be validated as supporting mission requirement components. Approved standards should support defined mission requirements through system functional requirements. Traceability is important. Ad hoc requirement definition is not inherently bad, but ad hoc requirements that cannot be validated in terms of specific mission activity support should be considered invalid for further exploration. Similarly, if a registered standard cannot be shown to support at least one defined mission requirement, it should be deregistered as an approved standard. Data models and object models remain players in this arena but should become models of approved standards tied to defined requirements. They should be composed from the bottom up using validated standards.

Conclusion

A change toward competitive registration of standards and bottom-up standards model development and away from dictated single data structure models would result in a data standardization program that makes sense. Standards would be defined in usable form. Standards could be traced to mission-based requirements. Most important, standards would be used to enhance communication between systems without the side effects of retarded development and increased cost. ◆

About the Authors

Gary A. Ham is a principal research scientist for Battelle Memorial Institute, National Security Division, Information Systems Engineering and Process Modernization Department in Arlington, Va. A former Marine Corps comptroller and Naval Academy computer science instructor, he is currently researching value metrics definition processes to support object-oriented requirements analysis and design of DoD systems. He has a bachelor's degree in economics from Whitman College in Walla Walla, Wash. and a master's degree (with distinction) in information systems management from the Naval Postgraduate School in Monterey, Ala. He is currently a doctoral student in information technology at George Mason University in Fairfax, Va.

Battelle Memorial Institute
2101 Wilson Blvd., Suite 800
Arlington, VA 22201-3008
Voice: 703-575-1072
Fax: 703-820-8817
E-mail: gham@erols.com

Douglas D. Mann is a senior research scientist at Battelle Memorial Institute in Arlington, Va. He has 27 years experience with Control Data Corporation as a consultant in data management. He has also worked at InfoSpan Corporation implementing repository systems using the Information Resource Dictionary System standard. He is currently on the L8 – Data Representation Standards subcommittee as editor of the draft ANSI standard X3.285 – Metamodel for the Management of Sharable Data. He also has prior experience as a member of the H4 – Open Repository Standards subcommittee. He is currently involved in the implementation of a shared data registry for the U.S. Environmental Protection Agency.

Battelle Memorial Institute
2101 Wilson Blvd., Suite 800
Arlington, VA 22201-3008
Voice: 703-575-0333
Fax: 703-575-0383
E-mail: mannd@battelle.org

References

1. DoD Directive 8320.1, "DoD Data Administration," September 1991.
2. Public Law: 104-191, "Health Insurance Portability and Accountability Act of 1996," ftp://ftp.loc.gov/pub/thomas/c104/h3103.enr.txt.
3. "Defense IRM Poor Implementation of Management Controls Has Put Migration Strategy at Risk," U.S. General Accounting Office, Report to the Ranking Minority Member, Committee on Governmental Affairs, U.S. Senate, GAO/AIMD-98-5, October 1997.
4. Information Resource Dictionary System – Information Model (working paper of X3H4 Information Resource

Dictionary System Committee, X3H4 92/0xx), April 8, 1992.
5. ISO/IEC 11179, "Information Technology – Specification and Standardization of Data Elements," ftp://sdct-sunsrv1.ncsl.nist.gov/x3l8/11179.
6. dpANS X3.285, "Metamodel for the Management of Sharable Data," ftp://sdct-sunsrv1.ncsl.nist.gov/x3l8/x3l8docs/x3.285.
7. Draft Technical Report, "Concept of Operations for a Data Element Registry," ftp://sdct-sunsrv1.ncsl.nist.gov/x3l8/x3l8docs/drconops.rtf.
8. Defense Information Infrastructure SHADE Capstone Document, Version 1.0, July 11,1996.

Notes

1. Http://www-datadmn.itsi.disa.mil provides information on current DoD data standardization policy, including access to the most current update of the DoD Data Model.
2. Official policy requires data models that are based on the structure of the DoD model, but the policy is not specific as to the detail required. Standards constructed using standard naming conventions and representations can theoretically be approved without imposing the rigidity of a standard model. Unfortunately, all functional areas with which we are familiar (four out of more than a dozen) have interpreted both written and verbal guidance from DoD to require detailed standard models. Furthermore, we have witnessed potential standards submitted without detailed compatibility turned down as standards in two functional areas.
3. The term *coupling* refers to the situation in which one module in a system shares internal information with another module to the extent that modification to either automatically requires modification to both. In programming, global variables used in multiple procedures "couple" the procedures together for maintenance purposes. We can say that data coupling occurs when disparate modules directly access a database structure. In such cases, changes to the database required in support of one module affect all other modules that access the same data. With modular encapsulation, change can be limited to the interface level, which reduces the degree of maintenance required.
4. To be fair, the two-digit year standard was not so "badly chosen" at its origin. With memory space at a premium, it was a good idea at the time. But "time" is the operative word here. Over time, good standards can become bad standards. Forcing data standardization into the bowels of otherwise disparate systems makes the inevitable correction process much more difficult.
5. Specialized languages, human and computer, may be more useful for specialized purposes (encapsulated purposes). They will still require translation into a more generalized "standard" if communication with outside people (or systems) is required.
6. "Ain't" is a well-understood, generalized representation for a concept whose more preferred representations are "am not," "are not," and "is not." As a generalization, ain't is a more "standard" term than any of its substitutes.
7. The development of human language constructs is not top down, either. The only known human language constructed from the top down is Esperanto. Although there is an Esperanto language authority, there are no native Esperanto speakers, and adoption of Esperanto has gone essentially nowhere. To adopt standards that are not already in general use in some form is likely to achieve the same lack of success.

CROSSTALK

Sponsor	***Lt. Col. Joe Jarzombek*** 801-777-2435 DSN 777-2435 jarzombj@software.hill.af.mil
Publisher	***Reuel S. Alder*** 801-777-2550 DSN 777-2550 publisher@stsc1.hill.af.mil
Managing Editor	***Forrest Brown*** 801-777-9239 DSN 777-9239 managing_editor@stsc1.hill.af.mil
Senior Editor	***Sandi Gaskin*** 801-777-9722 DSN 777-9722 senior_editor@stsc1.hill.af.mil
Graphics and Design	***Kent Hepworth*** 801-775-5555 ext. 3027 graphics@stsc1.hill.af.mil
Associate Editor	***Lorin J. May*** 801-775-5555 ext. 3026 backtalk@stsc1.hill.af.mil
Editorial Assistant	***Bonnie May*** 801-775-5555 ext. 3022 customer_service@stsc1.hill.af.mil
Features Coordinator	***Heather Winward*** 801-775-5555 ext. 3023 features@stsc1.hill.af.mil
Customer Service	***Barbara McDonald*** 801-777-8045 DSN 777-8045 custserv@software.hill.af.mil
Fax	801-777-8069 DSN 777-8069
STSC On-Line	http://www.stsc.hill.af.mil
***CROSSTALK* On-Line**	http://www.stsc.hill.af.mil/Crosstalk/crostalk.html
ESIP On-Line	http://www.esip.hill.af.mil

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address:

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

E-mail: custserv@software.hill.af.mil
Voice: 801-777-8045 DSN 777-8045
Fax: 801-777-8069 DSN 777-8069

Editorial Matters: Correspondence concerning *Letters to the Editor* or other editorial matters should be sent to the same address listed above to the attention of *CROSSTALK Editor* or send directly to the senior editor via the E-mail address also listed above.

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 *Guidelines for CROSSTALK Authors*, available upon request. We do 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: All product names referenced in this issue are trademarks of their companies. The mention of a product or business in *CROSSTALK* does not constitute an endorsement by the Software Technology Support Center (STSC), the Department of Defense, or any other government agency. The opinions expressed represent the viewpoints of the authors and are not necessarily those of the Department of Defense.

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 On-Line Services: STSC On-Line Services can be reached on the Internet. World Wide Web access is at http://www.stsc.hill.af.mil. The STSC maintains a Gopher server at gopher://gopher.stsc.hill.af.mil. Its ftp site may be reached at ftp://ftp.stsc.hill.af.mil. The Lynx browser or gopher server can also be reached using telnet at bbs.stsc.hill.af.mil or by modem at 801-774-6509 or DSN 775-3602. Call 801-777-7989 or DSN 777-7989 for assistance, or E-mail to portr@software.hill.af.mil.

Publications Available: The STSC provides various publications at no charge to the defense software community. Fill out the Request for STSC Services card in the center of this issue and mail or fax it to us. If the card is missing, call Customer Service at the numbers shown above, and we will send you a form or take your request by phone. The STSC sometimes has extra paper copies of back issues of *CROSSTALK* free of charge. If you would like a copy of the printed edition of this or another issue of *CROSSTALK*, or would like to subscribe, please contact the customer service address listed above.

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 that will improve the quality of their software products, their efficiency in producing them, and their ability to accurately predict the cost and schedule of their delivery. *CROSSTALK* is assembled, printed, and distributed by the Defense Automated Printing Service, Hill AFB, UT 84056. *CROSSTALK* is distributed without charge to individuals actively involved in the defense software development process.

Get Your Premium's Worth

I was struggling to find a topic for this column until a few weeks ago when it hit me—"it" being a big black truck. As the other guy's insurance adjuster seems to view it, the poor guy was minding his own business, heading straight through a red light from the right turn lane, when I ran a green light and rammed into the front of his truck with the side of my car. (This couldn't have upset the guy too much, however, since he didn't feel the need to stop after the accident.) But there is a plus side to my ongoing debate with the insurance company—I no longer feel that lawyers are the lowest form of life. In fact, for the first time in my life, I'm even doing business with one.

The point is, you often have to fight to get the insurance reimbursement you deserve, even if you are clearly in the right. This is especially the case with work-related injuries. Insurance companies are likely to only take macho-sounding injuries seriously ("torso bisected by acetylene torch") while downplaying the real risks you are likely to encounter at your desk job. Below is just a small sampling of documented hazards likely to be ignored by your employer's insurance policy.

Eyeball hyperextension – occurs when a colleague makes a sincere but unusually clueless statement such as "All we need now is the unanimous and continued support of upper management …," causing you to involuntary roll your eyes violently. The strain can be especially severe if there's a follow-up statement such as "… and as long as we receive adequate funding throughout the life of the project."

Spinal reverseitis – caused by bending over backward too far and too often for irritating clients or sponsors. The same events also often provoke involuntary facial ticks, a nerve-related disorder seen after hours of having one part of the brain force a cooperative smile while the rest of the brain sends signals for a bitter sneer. Also frequently seen in concert with various mouth injuries that result from biting one's lips or tongue to hold in politically unwise comments.

CRL ALT DEL strain – an inflammation of the right index finger and left middle and index fingers caused by having to reboot too frequently due to freeze-ups in the computer or network. Can be cured only with daily therapy and prolonged exposure to a Mac or UNIX environment.

Athlete's scalp – a fungal rash caused by individuals who step on your head on the way up the promotion ladder, then keep one foot there to prevent your ascension. Irritation can persist for years, only to go away immediately when the irritant is removed due to a gloriously satisfying fiasco.

Irony poisoning – a blood illness believed to be caused by absorbing ink while clipping out multiple Dilbert comics to pin to your cubicle wall. Some, however, believe it is strictly an environmental disorder. Symptoms include increased sarcasm, guilt-free unproductivity, and the tendency to spend inordinate amounts of time blaming everything that bothers you, including El Niño, on your superiors.

Realignment whiplash – neck pain caused by having a new boss blast in and immediately try to change everything before being quickly reassigned. Fortunately, ill effects in most organizations have long been minimized through the widespread use of a special neck brace available from MSQ (Maintain Status Quo), Inc., maker of the popular *Process-Proof Vest*.

Bladder distension angina – occurs in morning meetings dominated by people who have much more time to kill—and much less coffee in their systems—than you. However, this disorder is often intentionally self-inflicted for afternoon meetings as it is preferable to involuntary nasal impact syndrome, a painful and embarrassing disorder caused by a combination of heavy lunch, boring subject matter, lights dimmed for presentations, and hard table surfaces.

The insurance industry won't take these disorders seriously until more of us file claims. However, they won't reimburse you without a "cost code" for your injury, and they'll create that code only when there is a statistical basis for one. Therefore, I recommend filing six or seven claims per day. But I have deep and abiding faith the industry will one day acknowledge these disorders; based on my experiences, they will then settle your claim faster than you can say "after continental drift pushes California within spitting distance of downtown Sydney." – Lorin May

Got an idea for BACKTALK? Send an E-mail to backtalk@stsc1.hill.af.mil