XML and the Second-Generation WEB

by Jon Bosak and Tim Bray

The combination of hypertext and a global Internet started a revolution. A new ingredient, XML, is poised to finish the job

Give people a few hints, and they can figure out the rest. They can look at this page, see some large type followed by blocks of small type and know that they are looking at the start of a magazine article. They can look at a list of groceries and see shopping instructions. They can look at some rows of numbers and understand the state of their bank account.

Computers, of course, are not that smart; they need to be told exactly what things are, how they are related and how to deal with them. Extensible Markup Language (XML for short) is a new language designed to do just that, to make information self-describing. This simple-sounding change in how computers communicate has the potential to extend the Internet beyond information delivery to many other kinds of human activity. Indeed, since XML was completed in early 1998 by the World Wide Web Consortium (usually called the W3C), the standard has spread like wildfire through science and into industries ranging from manufacturing to medicine.

The enthusiastic response is fueled by a hope that XML will solve some of the Web's biggest problems. These are widely known: the Internet is a speed-of-light network that often moves at a crawl; and although nearly every kind of information is available on-line, it can be maddeningly difficult to find the one piece you need.

Both problems arise in large part from the nature of the Web's main language,

XML BRIDGES the incompatibilities of computer systems, allowing people to search for and exchange scientific data, commercial products and multilingual documents with greater ease and speed.

ILLUSTRATIONS BY BRUCE ROSCH

<Together XML and XSL allow publishers to pour a publication into myriad forms—write once and publish everywhere. />

HTML (shorthand for Hypertext Markup Language). Although HTML is the most successful electronic-publishing language ever invented, it is superficial: in essence, it describes how a Web browser should arrange text, images and pushbuttons on a page. HTML's concern with appearances makes it relatively easy to learn, but it also has its costs.

One is the difficulty in creating a Web site that functions as more than just a fancy fax machine that sends documents to anyone who asks. People and companies want Web sites that take orders from customers, transmit medical records, even run factories and scientific instruments from half a world away. HTML was never designed for such tasks.

So although your doctor may be able to pull up your drug reaction history on his Web browser, he cannot then e-mail it to a specialist and expect her to be able to paste the records directly into her hospital's database. Her computer would not know what to make of the information, which to its eyes would be no more intelligible than <H1>blah blah </H1> <BOLD>blah blah blah </BOLD>. As programming legend Brian Kernighan once noted, the problem with "What You See Is What You Get" is that what you see is all you've got.

Those angle-bracketed labels in the example just above are called tags. HTML has no tag for a drug reaction, which highlights another of its limitations: it is inflexible. Adding a new tag involves a bureaucratic process that can take so long that few attempt it. And yet every application, not just the interchange of medical records, needs its own tags.

Thus the slow pace of today's on-line bookstores, mail-order catalogues and other interactive Web sites. Change the quantity or shipping method of your order, and to see the handful of digits that have changed in the total, you must ask a distant, overburdened server to send you an entirely new page, graphics and all. Meanwhile your own high-powered machine sits waiting idly, because it has only been told about <H1>s and <BOLD>s, not about prices and shipping options.

Thus also the dissatisfying quality of Web searches. Because there is no way to mark something as a price, it is effectively impossible to use price information in your searches.

Something Old, Something New

The solution, in theory, is very simple: use tags that say what the information is, not what it looks like. For example, label the parts of an order for a shirt not as boldface, paragraph, row and column—what HTML offers—but as price, size, quantity and color. A program can then recognize this document as a customer order and do whatever it needs to do: display it one way or display it a different way or put it through a bookkeeping system or make a new shirt show up on your doorstep tomorrow.

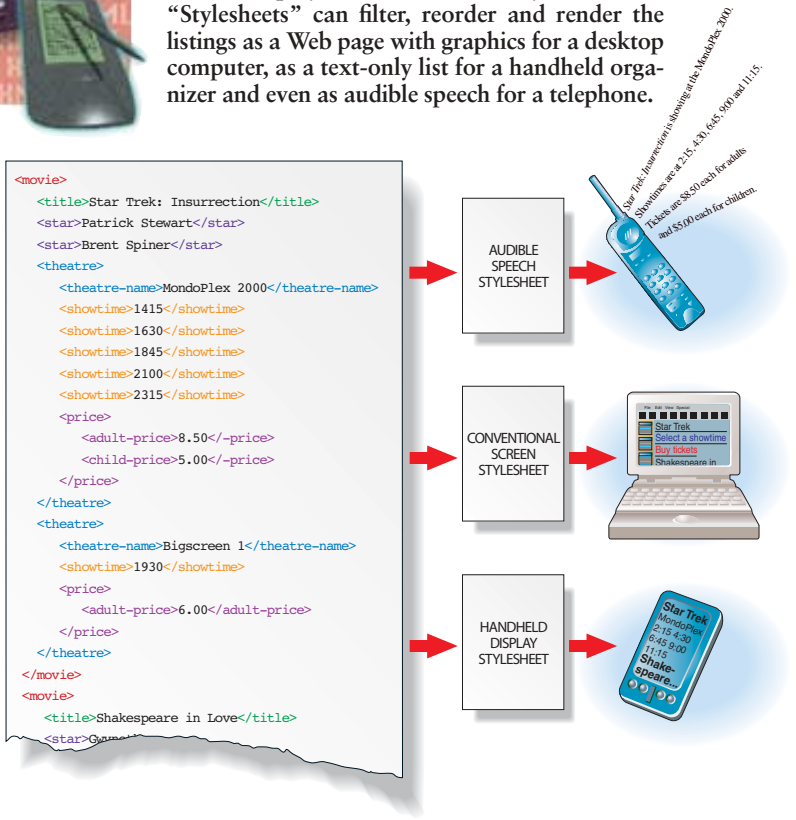
We, as members of a dozen-strong W3C working group, began crafting such a solution in 1996. Our idea was powerful but not entirely original. For generations, printers scribbled notes on manuscripts to instruct the typesetters. This "markup" evolved on its own until 1986, when, after decades of work, the International Organization for Standardization (ISO) approved a system for the creation of new markup languages.

Named Standard Generalized Markup Language, or SGML, this language for describing languages—a metalanguage—has since proved useful in many large publishing applications. Indeed, HTML was defined using SGML. The only problem with SGML is that it is *too* general—full of clever features designed to minimize keystrokes in an era when every byte had to be accounted for. It is more complex than Web browsers can cope with.

Our team created XML by removing frills from SGML to arrive at a more streamlined, digestible metalanguage. XML consists of rules that anyone can follow to create a markup language from scratch. The rules ensure that a single compact program, often called a parser, can process all these new languages.

Consider again the doctor who wants to e-mail your medical record to a spe-

MARKED UP WITH XML TAGS, one file—containing, say, movie listings for an entire city—can be displayed on a wide variety of devices. "Stylesheets" can filter, reorder and render the listings as a Web page with graphics for a desktop computer, as a text-only list for a handhold organizer and even as audible speech for a telephone.



LAURIE GRACE

cialist. If the medical profession uses XML to hammer out a markup language for encoding medical records—and in fact several groups have already started work on this—then your doctor's e-mail could contain <patient> <name> blah blah </name> <drug-allergy> blah blah blah </drug-allergy> </patient>. Programming any computer to recognize this standard medical notation and to add this vital statistic to its database becomes straightforward.

Just as HTML created a way for every computer user to read Internet documents, XML makes it possible, despite the Babel of incompatible computer systems, to create an Esperanto that all can read and write. Unlike most computer data formats, XML markup also makes sense to humans, because it consists of nothing more than ordinary text.

The unifying power of XML arises from a few well-chosen rules. One is that tags almost always come in pairs. Like parentheses, they surround the text to which they apply. And like quotation marks, tag pairs can be nested inside one another to multiple levels.

The nesting rule automatically forces a certain simplicity on every XML document, which takes on the structure known in computer science as a tree. As with a genealogical tree, each graphic and bit of text in the document represents a parent, child or sibling of some other element; relationships are unambiguous. Trees cannot represent every kind of information, but they can represent most kinds that we need computers to understand. Trees, moreover, are extraordinarily convenient for programmers. If your bank statement is in the form of a tree, it is a simple matter to write a bit of software that will reorder the transactions or display just the cleared checks.

Another source of XML's unifying strength is its reliance on a new standard called Unicode, a character-encoding system that supports intermingling of text in all the world's major languages. In HTML, as in most word processors, a document is generally in one particular language, whether that be English or Japanese or Arabic. If your software cannot read the characters of that language, then you cannot use the document. The situation can be even worse: software made for use in Taiwan often cannot read mainland-Chinese texts because of incompatible encodings. But software that reads XML properly can deal with any combination of any of these character sets. Thus, XML enables exchange of information not only between different computer systems but also across national and cultural boundaries.

An End to the World Wide Wait

As XML spreads, the Web should become noticeably more responsive. At present, computing devices connected to the Web, whether they are powerful desktop computers or tiny pocket planners, cannot do much more than get a form, fill it out and then swap it back and forth with a Web server until a job is completed. But the structural and semantic information that can be added with XML allows these devices to do a great deal of processing on the spot. That not only will take a big load off Web servers but also should reduce network traffic dramatically.

To understand why, imagine going to an on-line travel agency and asking for all the flights from London to New York on July 4. You would probably receive a list several times longer than your screen could display. You could shorten the list by fine-tuning the departure time, price or airline, but to do that, you would have to send a request across the Internet to the travel agency and wait for its answer. If, however, the long list of flights had been sent in XML, then the travel agency could have sent a small Java program along with the flight records that you could use to sort and winnow them in microseconds, without ever involving the server. Multiply this by a few million Web users, and the global efficiency gains become dramatic.

As more of the information on the Net is labeled with industry-specific XML tags, it will become easier to find exactly what you need. Today an Internet search for "stockbroker jobs" will inundate you with advertisements but probably turn up few job listings—most will be hidden inside the classified ad services of newspaper Web sites, out of a search robot's reach. But the Newspaper Association of America is even now building an XML-based markup language for classified ads that promises to make such searches much more effective.

Even that is just an intermediate step. Librarians figured out a long time ago that the way to find information in a hurry is to look not at the information itself but rather at much smaller, more focused sets of data that guide you to the useful sources: hence the library card catalogue. Such information about information is called metadata.

From the outset, part of the XML project has been to create a sister standard for metadata. The Resource Description Framework (RDF), finished this past February, should do for Web data what catalogue cards do for library books. Deployed across the Web, RDF metadata will make retrieval far faster and more accurate than it is now. Because the Web has no librarians and every Webmaster wants, above all else, to be found, we expect that RDF will achieve a typically astonishing Internet growth rate once its power becomes apparent.

There are of course other ways to find things besides searching. The Web is after all a "hypertext," its billions of pages connected by hyperlinks—those underlined words you click on to get whisked from one to the next. Hyperlinks, too, will do more when powered by XML. A standard for XML-based hypertext, named XLink and due later this year from the W3C, will allow you to choose from a list of multiple destinations. Other kinds of hyperlinks will insert text or images right where you click, instead of forcing you to leave the page.

Perhaps most useful, XLink will enable authors to use indirect links that point to entries in some central database rather than to the linked pages themselves. When a page's address changes, the author will be able to update all the links that point to it by editing just one database record. This should help eliminate the familiar "404 File Not Found" error that signals a broken hyperlink.

The combination of more efficient processing, more accurate searching and more flexible linking will revolutionize

< XML enables exchange of information not only between different computer systems but also across national and cultural boundaries. />

the structure of the Web and make possible completely new ways of accessing information. Users will find this new Web faster, more powerful and more useful than the Web of today.

Some Assembly Required

Of course, it is not quite that simple. XML does allow anyone to design a new, custom-built language, but designing *good* languages is a challenge that should not be undertaken lightly. And the design is just the beginning: the meanings of your tags are not going to be obvious to other people unless you write some prose to explain them, nor to computers unless you write some software to process them.

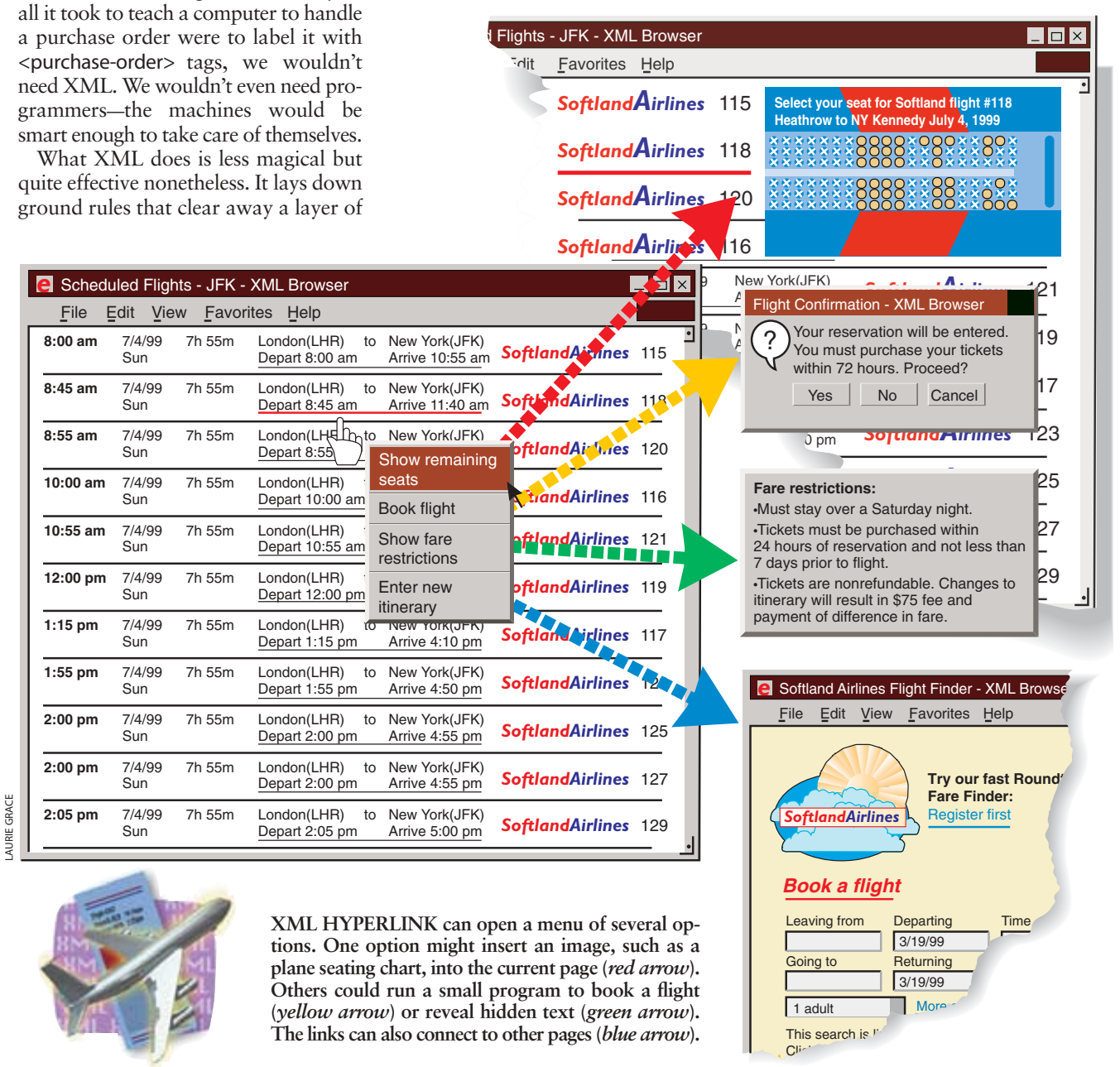
A moment's thought reveals why. If all it took to teach a computer to handle a purchase order were to label it with <purchase-order> tags, we wouldn't need XML. We wouldn't even need programmers—the machines would be smart enough to take care of themselves.

What XML does is less magical but quite effective nonetheless. It lays down ground rules that clear away a layer of programming details so that people with similar interests can concentrate on the hard part—agreeing on how they want to represent the information they commonly exchange. This is not an easy problem to solve, but it is not a new one, either.

Such agreements will be made, because the proliferation of incompatible computer systems has imposed delays, costs and confusion on nearly every area of human activity. People want to share ideas and do business without all having to use the same computers; activity-specific interchange languages go a long way toward making that possible. Indeed, a shower of new acronyms ending in "ML" testifies to the inventiveness unleashed by XML in the sciences, in business and in the scholarly disciplines [*see box on opposite page*].

Before they can draft a new XML language, designers must agree on three things: which tags will be allowed, how tagged elements may nest within one another and how they should be processed. The first two—the language's vocabulary and structure—are typically codified in a Document Type Definition, or DTD. The XML standard does not compel language designers to use DTDs, but most new languages will probably have them, because they make it much easier for programmers to write software that understands the markup and does intelligent things with it.

Programmers will also need a set of guidelines that describe, in human lan-



LAURIE GRACE

XML HYPERLINK can open a menu of several options. One option might insert an image, such as a plane seating chart, into the current page (*red arrow*). Others could run a small program to book a flight (*yellow arrow*) or reveal hidden text (*green arrow*). The links can also connect to other pages (*blue arrow*).

guage, what all the tags mean. HTML, for instance, has a DTD but also hundreds of pages of descriptive prose that programmers refer to when they write browsers and other Web software.

A Question of Style

For users, it is what those programs do, not what the descriptions say, that is important. In many cases, people will want software to display XML-encoded information to human readers. But XML tags offer no inherent clues about how the information should look on screen or on paper.

This is actually an advantage for publishers, who would often like to "write once and publish everywhere"—to distill the substance of a publication and then pour it into myriad forms, both printed and electronic. XML lets them do this by tagging content to describe its meaning, independent of the display medium. Publishers can then apply rules organized into "stylesheets" to reformat the work automatically for various devices. The standard now being developed for XML stylesheets is called the Extensible Stylesheet Language, or XSL.

The latest versions of several Web browsers can read an XML document, fetch the appropriate stylesheet, and use it to sort and format the information on the screen. The reader might never know that he is looking at XML rather than HTML—except that XML-based sites run faster and are easier to use.

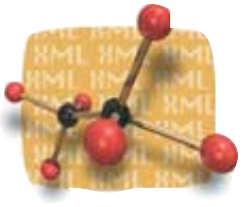
People with visual disabilities gain a free benefit from this approach to publishing. Stylesheets will let them render XML into Braille or audible speech. The advantages extend to others as well: commuters who want to surf the Web in their cars may also find it handy to have pages read aloud.

Although the Web has been a boon to science and to scholarship, it is commerce (or rather the expectation of future commercial gain) that has fueled its lightning growth. The recent surge in retail sales over the Web has drawn much attention, but business-to-business commerce is moving on-line at least as quickly. The flow of goods through the manufacturing process, for example, begs for automation. But schemes that rely on complex, direct program-to-program interaction have not worked well in practice, because they depend on a uniformity of processing that does not exist.

For centuries, humans have successfully done business by exchanging standardized documents: purchase orders, invoices, manifests, receipts and so on. Documents work for commerce because they do not require the parties involved to know about one another's internal procedures. Each record exposes exactly what its recipient needs to know and no more. The exchange of documents is probably the right way to do business on-line, too. But this was not the job for which HTML was built.

XML, in contrast, was designed for document exchange, and it is becoming clear that universal electronic commerce will rely heavily on a flow of agreements, expressed in millions of XML documents pulsing around the Internet.

Thus, for its users, the XML-powered Web will be faster, friendlier and a better place to do business. Web site designers, on the other hand, will find it more demanding. Battalions of programmers will be needed to exploit new XML languages to their fullest. And although the day of the self-trained Web hacker is not yet over, the species is endangered. Tomorrow's Web designers will need to be versed not just in the production of words and graphics but also in the construction of multilayered, interdependent systems of DTDs, data trees, hyperlink structures, metadata and stylesheets—a more robust infrastructure for the Web's second generation. SA



New Languages for Science

XML offers a particularly convenient way for scientists to exchange theories, calculations and experimental results. Mathematicians, among others, have long been frustrated by Web browsers' ablity to display mathematical expressions only as pictures. **MathML** now allows them to insert equations into their Web pages with a few lines of simple text. Readers can then paste those expressions directly into algebra software for calculation or graphing.

Chemists have gone a step further, developing new browser programs for their XML-based **Chemical Markup Language** (CML) that graphically render the molecular structure of compounds described in CML Web pages. Both CML and **Astronomy Markup Language** will help researchers sift quickly through reams of journal citations to find just the papers that apply to the object of their study. Astronomers, for example, can enter the sky coordinates of a galaxy to pull up a list of images, research papers and instrument data about that heavenly body.

XML will be helpful for running experiments as well as analyzing their results. National Aeronautics and Space Administration engineers began work last year on **Astronomical Instrument ML** (AIML) as a way to enable scientists on the ground to control the SOFIA infrared telescope as it flies on a Boeing 747. AIML should eventually allow astronomers all over the world to control telescopes and perhaps even satellites through straightforward Internet browser software.

Geneticists may soon be using **Biosequence ML** (BSML) to exchange and manipulate the flood of information produced by gene-mapping and gene-sequencing projects. A BSML browser built and distributed free by Visual Genomics in Columbus, Ohio, lets researchers search through vast databases of genetic code and display the resulting snippets as meaningful maps and charts rather than as obtuse strings of letters.

—The Editors

The Authors

JON BOSAK and TIM BRAY played crucial roles in the development of XML. Bosak, an on-line information technology architect at Sun Microsystems in Mountain View, Calif., organized and led the World Wide Web Consortium working group that created XML. He is currently chair of the W3C XML Coordination Group and a representative to the Organization for the Advancement of Structured Information Standards. Bray is co-editor of the *XML 1.0* specification and the related *Namespaces in XML* and serves as co-chair of the W3C XML Syntax Working Group. He managed the New Oxford English Dictionary Project at the University of Waterloo in 1986, co-founded Open Text Corporation in 1989 and launched Textuality, a programming firm in Vancouver, B.C., in 1996.