The Shortest Path to the Future Web

Danny Ayers • *Independent Consultant*

The W3C's motto is "Leading the Web to Its Full Potential." This seems a reasonable role for it to aspire to, but it raises at least two questions: can any organization actually lead the Web, and what is its full potential? I believe these two questions are tightly interrelated, and as far as any answers are available, it's the Web itself that offers them.

Clearly, a complete picture of the Web's full potential should consider its human impact, not least because people are the Web's most significant components. However, I won't even attempt to address those issues here. From a technical standpoint, on the other hand, we can identify and analyze aspects of the Web with improvement in mind. Broadly, these include the realms of human interface, services, and data. Unlike most journeys, the destination for the Web is unclear, but we might at least be able to consider a likely trajectory based on its past and present directions. I believe the most significant aspect of this trajectory will be a change from the Web as primarily a repository of interlinked documents to a more generalized, more dynamic system of interlinked data.

Semantic Web technologies offer a logical model through which application developers and, potentially, end users can integrate all kinds of data across the globe, irrespective of the data's domain. This approach appears technically feasible, but a significant gulf exists between the Web in its current form and a (Semantic) Web of Data. Currently, the typical Web developer is unlikely to have much familiarity with Semantic Web technologies. Current trends do indicate a tendency toward a Web of Data, but rather than leaping across the document–data chasm, the Web is taking baby steps along a less direct path.

The Traditional Web

The information world before the Web was hardly void or without form; the Internet had already become a communication medium, with email as its most significant application. Distributed data was available, accessible through transfer protocols such as Gopher and ftp, and indexed through search protocols such as Wide Area Information Servers (WAIS) and Archie. The World Wide Web brought a simple hypertext markup language (HTML) with the support of a simple protocol: HTTP. Together, these provide a way for network users to retrieve documents, which in turn can contain references to other documents — that is, HTML links. This combination enables distributed cross-referencing and navigation across the information space.

One key to the Web's success is its simplicity — once a handful of clients and servers existed, it became unstoppable. It's easy to create an HTML document, especially as most HTTP clients (browsers) are very forgiving of errors. More fundamentally, the Web is successful because there is only *one Web*. In other words, at its core, the Web offers a uniform interface to distributed data (although the potential for extension is unlimited). Above all is the abstract notion of the uniformly identifiable resource — every document is identified with a URI.

The Web is a network, a communications infrastructure built from computers of all sizes, with information distributed throughout it. However, huge gains in communication make it easy to forget that computers are best at computing. Plenty of work is under way on computational grid systems that can exploit this capability in networks, but few of these are designed to work directly on the Web. One reason for this is that most practical, useful computing is essentially data processing. For a network to act as a computer, processable data must be readily available. Little current Web material lends itself to this.

 Published by the IEEE Computer Society 1089-7801/06/$20.00 © 2006 IEEE

Computers are well equipped to deal with all kinds of data: numeric, textual, visual images, media objects, and so on. Yet the current Web is primarily a document repository. These documents might contain numbers, and HTTP can deliver multiple media types, but there is little granularity in access to anything but text, and even with text, content addressability has been limited until relatively recently. But change is under way.

Revising the Web

Web technologies are undergoing a resurgence in creativity, popularly labeled "Web 2.0." The term in itself is little more than jargon, but tangible initiatives exist under its umbrella. The most visible is the rediscovery of client-side Javascript and its capability, when used alongside (X)HTML and HTTP, for improving the user experience. Taken together, this toolset has been rebranded Asynchronous JavaScript and XML (AJAX). In many cases, it offers little more than decoration and minor enhancements to interaction — definitely improvements, but nothing seismic. However, one class of applications, known as *mashups*, do point to something deeper. A mashup combines data or content from more than one online source. A typical example might be the integration of a system that lists public events with a system that generates geographic maps to produce a hybrid view of the events marked on a map. The recent explosion of RSS/Atom syndication opens the door to a similar kind of recombinant data integration. In syndication, the content's essentials are, in effect, lifted from the traditional Web site or homepage context and published without styling information but with enhanced metadata (title, date, links, and so on), which makes it possible for end-user tools to mash up the content with material from other sources.

The Future Is Semantic?

I started by quoting the W3C's motto, and one of its initiatives is directly relevant here: the Semantic Web initiative aims to enable adding first-class data to the current Web in a uniformly addressable and machine-processable fashion. Central to Semantic Web technologies is the Resource Description Framework (RDF). Effectively a data model built on logical foundations, RDF can support fairly sophisticated knowledge representation through RDF Schema (RDFS) and the Web Ontology Language (OWL). Although many of the ideas behind the Semantic Web have their basis in old AI knowledge representation, the Semantic Web is designed as an extension of the existing Web. The "resource" in RDF is the uniformly identifiable resource. Whereas typically on the Web URIs identify human-readable documents, the Semantic Web goes further — URIs globally identify any thing, real or virtual. Real-world and conceptual systems can be modeled on the Web, not just as documents or raw data hidden in database tables.

However, much Semantic Web work is avant-garde, quite far from the mainstream, so the gulf between the Web of Documents and the (Semantic) Web of Data remains. Bridging this gulf requires a paradigm shift, and in many ways, it would be revolutionary. But risk is involved in any revolution, and in the real world, many historical cases exist in which well-motivated revolutions ended in disaster. So perhaps what we need is a *velvet* paradigm shift. In fact, one might already be happening.

Increments vs. Leaps

Incremental development is a recognized approach to writing software, particularly promoted around Extreme Programming (XP). It involves frequent releases with small changes rather than to major releases over a longer timescale. The advantages cited include

- increased control over the project,
- feedback that can get near-immediate responses, and
- the ability to continuously ensure that the system works (when the approach is applied in concert with fine-grained regression tests).

The tight feedback loop in XP ensures that the direction taken is the one required. Continuous correction keeps the project on track, even if the track itself changes direction. This approach might have an analog at a larger scale.

HTTP can deliver multiple media types, but there is little granularity in access to anything but text.

Many Web 2.0 notions such as tagging, aggregation, filtering, and content ranking are now appearing in mainstream sites. One characteristic these ideas share is a good short-term cost–benefit ratio. Developers can incorporate them into existing sites as relatively small developmental increments, but they represent immediate visible additions to a site's feature set. Yet, in practice, such facilities are usually added not as full-blown Web extensions using specifications like those of the Semantic Web stack but as extensions to individual applications. Where increased data modeling is needed, the typical developer's incremental path will come from his or her local system, not that of the Web at large. Developers will tend to implement an application feature, such as associating keyword tags with content, on top of the existing back-end storage (usually a relational data-

base) in line with their local data models (often expressed in an object-oriented programming language). A side effect of this is that the data can't be exposed directly to the rest of the Web because the language it's expressed in makes sense only within the local context. Interfaces that developers make available are usually created around a custom domain-specific language that reflects the internal model or, at the other extreme, around the lowest common denominator of HTML.

Three Strategies

Every computer system deals with data locally, so the problem isn't actually in creating the data but in finding the appropriate language in which to make it available on the Web. Current practice is generally to use HTML, but on its own, this is severely limited when it comes to machine reuse. From the incremental development viewpoint, at least three general strategies exist for exposing that data.

First, developers can add Semantic Web-oriented interfaces to existing systems — places to receive and provide RDF over HTTP, along with generic query interfaces using the SPARQL protocol and RDF Query Language. Given the tools and libraries now available, constructing the modeling and wiring needed for bridges between the Semantic Web and local data is relatively straightforward. The hard parts are usually inherently hard problems, such as determining and implementing appropriate access controls and looking after scalability with concurrent access. Unfortunately, although the addition of dedicated interfaces is heading in the right direction for the Semantic Web — and can be achieved as an increment to existing systems — the immediate benefits to a company or organization's business are far from clear. Without near-term, tangible gain there's little to motivate adding Semantic Web interfaces. This isn't to say there won't be such a gain but it's less than easy to chalk up on the spreadsheet.

Another small step from the current Web to Web 2.0 is to embed machine-readable data in existing HTML content. This can happen in various ways, the poster child being the microformats initiative. Essentially, microformats are a set of conventions that enable machine-friendly access to information in human-oriented markup (typically HTML). The conventions exploit the markup's existing semantics as well as structural relationships such as linking and element nesting. Additional semantics are layered on top using domain-specific controlled vocabularies in standard HTML constructs (for example, <a href="http://example.org" rel="friend"> might link to a friend's homepage). The HTML specification describes metadata profiles, and microformat documents ideally will be associated (via a profile attribute in the document's `<head>`) to the URIs of the vocabularies they contain. Using profiles in this way is the difference between scraping and deterministically extracting data from a document.

A third strategy toward a Web of Data is to return to Semantic Web technologies' roots and enrich human-readable content with machine-readable metadata. This strategy differs slightly from embedding data in that the data provided needn't physically be part of the document (for example, references could be in separate BibTex files), although it will always be about the document. Metadata is data by definition, but in the context of documents, it's a stage removed from whatever the document is describing.

One Web

Most Web developers aren't interested in a Semantic Web — what they want is to improve the user experience. Clearly, their most immediate concerns are local, largely pertaining to content management and user interface. But as Web systems diversify, it's increasingly possible for developers to take advantage of external systems and less traditional Web publishing techniques such as those with the Web 2.0 label — syndication, mashups, and microformats, for example. Key to all these techniques is data interoperability. The question is, interoperability at what level?

Traditionally on the Web, we've described real-world things and concepts only in a form designed for direct human consumption — that is, Web pages. The computer network is acting as a human–human communication system. The W3C's Dan Connolly has referred to the Web as "the minimum amount of distributed object technology necessary to get the job done" (see www.w3.org/People/Connolly/9703-web-apps-essay.html). The objects on the Web right now are primarily human-oriented documents, a far cry from the objects found in other software technologies.

But the Web already supports the expression of simple interdocument relationships through hyperlinks, and most documents are associated with significant amounts of potentially machine-readable information: authorship details, publisher information, subject classification, citations, revision history, and so on. Each of these facets leads to a wealth of data — for instance, the author's professional information, the publisher's catalogue, or real-world entities in the subject's scope. Document metadata is immediately useful on the current Web through indexing for search and navigation. Web 2.0 can use the metadata for systematic republication and mashups. But that metadata is only a whisker away from data that isn't necessarily associated with documents. To mangle Arthur C. Clarke's observation on technology and magic, any sufficiently advanced metadata is indistinguishable from first-class data.

But existing information can often be expressed in a form that lends itself to machine processing (which is one

motivation behind ideas like microformats). Once you have machine-processable data, you can have computer–computer communication with considerably less human intermediation. We can't deny computers' power in processing data in local systems, and nothing suggests that we can't apply such power on a global scale, given the connected platform the Web provides. The Semantic Web vision does offer a virtual destination built on the current Web, but it might appear out of reach. Yet many individually minor advances are possible, and the majority point in the same general direction as the Semantic Web.

This column's title could suggest that there is only one best path forward for the Web. I think one path begins with document metadata (as found around microcontent and syndication) and travels through the world of microformats and embedded data. A waypoint will be a Semantic Web that leverages these approaches, along with those offered by an environment more capable of managing first-class data directly. This is only one path, however, and it probably isn't the shortest (although because it's made of small steps, I suspect it will be well-traveled).

The Internet is a rich environment with billions of active agents. Natural selection, mutation, and genetic breeding of sorts all happen to software systems, together with a significantly higher proportion of "intelligent design" than found in the real world. The net effect is that many different evolutionary paths are being explored simultaneously, and several could lead to a better Web.

The notion of a Web of Data seems compelling to the point of inevitability right now, but that might change. When it comes to the W3C "leading the Web," at least as far as the Semantic Web initiative is concerned, it might be like leading a horse to water and having to wait for it to get thirsty. But what does seem fairly certain is that those paths that build on the Web's successful features (in particular, decentralization and interface uniformity) will probably be the easiest in the long run.

In this column I've given an overview of where I suspect the Web is headed over the next few years. In the next, I'll give some concrete examples of technologies that I believe provide evidence for this being a likely direction.

Danny Ayers is an independent developer, consultant, and author. His research interests are primarily around Semantic Web technologies. Ayers has coauthored ten books on programming, generally covering Web-related topics. He is chair of the Developers Track for WWW 2007. His Weblog is at http://dannyayers.com. Contact him at danny.ayers.ieee@gmail.com.