# The Dataweb: An Introduction to XDI

A White Paper for the OASIS XDI Technical Committee
January 20, 2004

Drummond Reed, Cordance
Geoffrey Strongin, AMD

## Abstract

XDI (XRI Data Interchange) is a new service for generalized distributed data sharing and mediation using XRIs (Extensible Resource Identifiers), a URI-compatible abstract identifier scheme developed by the OASIS XRI Technical Committee. The goal of XDI is to enable data from any data source to be identified, exchanged, linked, and synchronized into a machine-readable *dataweb* using XML documents just as content from any content source can linked into the human-readable Web using HTML documents today. Because the controls needed to mediate access and usage of shared data can be built right into every XDI link, the emergence of a global Dataweb has the potential to do for trusted data interchange what the Web did for open content exchange.

This white paper presents several examples of classic cross-domain data sharing problems, explains how the Dataweb model can provide a generalized solution, and describes the key objectives of the newly formed OASIS XDI Technical Committee (http://www.oasis-open.org/committees/xdi).

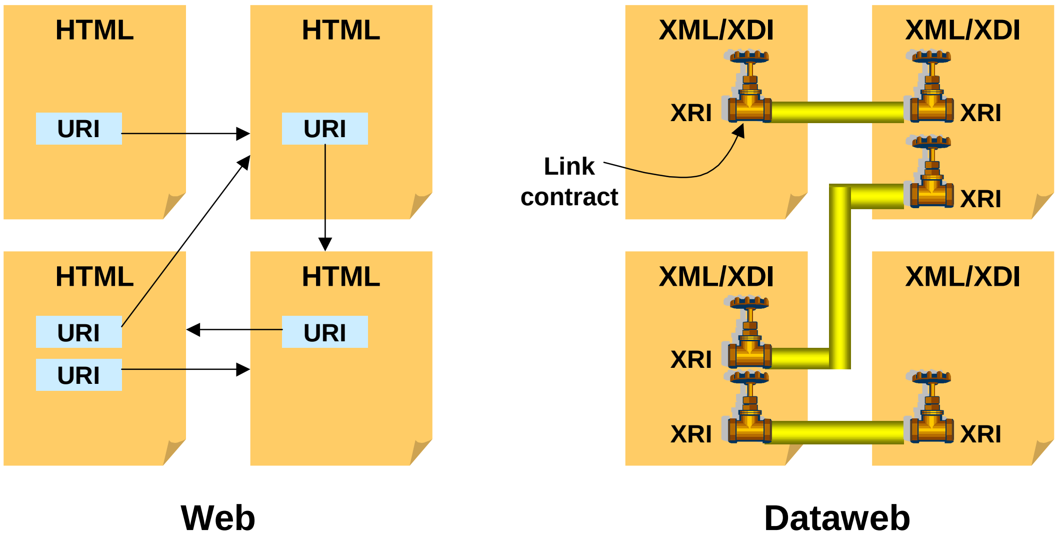

*Table of Contents*

# Introduction

> *"I would not give a fig for the simplicity this side of complexity, but I would give my life for the simplicity on the other side of complexity."*
>
> -- *Oliver Wendell Holmes, Jr.*

The Web has dramatically simplified information access. Never in history has content been so readily available or richly linked. Furthermore, this model for distributed systems has become so compelling that the next wave has already broken: Web services—the promise of distributed applications talking to each other in XML and SOAP as easily as Web browsers and servers talk in HTML and HTTP.

Yet with cross-domain applications comes another complex problem: the need to share instances of the same data across multiple domains, directories, databases, and applications. There are many facets to this challenge: identification, authentication, authorization, mediation, and synchronization; all problems that were minimized on the Web because it dealt primarily with *presentation* of data rather than *interchange* of data.

Recently a new approach to this set of needs has emerged called the *Dataweb*. The Dataweb combines the key principles of Web architecture with core concepts from Web services and the Semantic Web into a solution that can be "the simplicity on the other side of complexity" for distributed data sharing.

This white paper presents several common distributed data sharing challenges and then explains how the Dataweb model can be applied to solve them. It then provides more details about how the Dataweb model builds on principles of Web, Web Services, and Semantic Web architectures. It closes by summarizing the mission of the newly formed OASIS XDI Technical Committee.

# Three Examples of Distributed Data Sharing Challenges

When talking about generalized solutions, it can be very helpful to have concrete examples of the problems they address. This section presents three examples of well-known distributed data sharing challenges that have a daily impact on personal and business productivity.

## *Electronic Business Cards and Dynamic Address Books*

One of the oldest rituals in business is the exchange of business cards when forming new relationships. One of the second oldest rituals in business is trying to update this information across your entire contact network when it changes. It is a fact of business life: people move, and relationships change.

So it was inevitable that the rise of the Internet would spur the development of electronic business cards and "dynamic" address books—address books that accept new contacts and updates to contact data automatically, much the way Internet email servers send and receive messages automatically. In the last decade there have been no less than a dozen attempts to start Internet business card sharing networks. From PlanetAll to Plaxo, each of these has introduced a new system for exchanging and updating electronic business cards. And each has faced the same fundamental problem; like Internet email, they must become nearly ubiquitous before their full value can be realized.

Typically this means standardization, and there have been numerous efforts to standardize contact data representation and interchange formats. An entire industry consortium, Versit, came together to create the vCard, a set of IETF RFCs now managed by the Internet Mail Consortium[1]. More recently, the OASIS

[1] http://www.imc.org/pdi/

Customer Information Quality (CIQ) Technical Committee has been working since 2000 to develop XML vocabulary for describing customer and contact data.[2]

However many of the business card sharing networks use these open formats, so agreement on a data representation format has not been the gating factor for their success. Rather the challenge is to make electronic business cards *dynamic*—self-updating across the network. Doing this interoperably means addressing three additional dimensions of complexity:

1. *Agreement to a standard data interchange protocol*, so that dynamic address book clients and servers can all talk to each other the same way Web browsers and servers talk to each other in HTTP.
2. *Agreement to a common data synchronization model* for sending updates to shared contact data, similar to that which SyncML provides for mobile devices.
3. *Agreement to a common trust model* for cross-domain authentication and authorization so dynamic address book servers can propagate updates across trust domains.

With this Rubik's cube of interoperability challenges, is it any wonder that the problem of global business card sharing has not been solved?

## Calendar Sharing

Business card exchange generally happens only at the initiation of new relationships, or when contact data is updated. But coordination of events between the contacts is ongoing throughout the lifetime of a relationship and happens across all spheres of activity—work, home, school, friends, etc. Therefore any Internet-based solution to calendar sharing must be cross-domain just like Internet email and the Web.

On the dimension of data representation, calendar sharing is easier that business card sharing. The element of time is common to all applications and networks, and a great deal of value can be provided even with the most limited event description vocabulary (e.g., Subject, Location, Start Time, End Time, etc.) Many standard formats have been developed to express this data, including the Versit complement to vCard called vCal[3] and Microsoft's format for sharing of calendering data from MS Outlook.[4]

However even if data representation for shared calendaring is easier, the other three dimensions of dynamic data sharing requirements are not. To interoperate on the same scale as Internet email or the Web, calendar servers need an interoperable data interchange protocol, an interoperable and scalable synchronization model, and an interoperable trust model for protecting access to shared calendar data.

This last dimension is particularly crucial. Mere knowledge that a meeting or phone call is taking place between certain participants can be highly sensitive information. This aspect of calendar sharing intersects one of the hottest issues in the wireless industry: presence and availability management (PAM).[5] Allowing one wireless user to instantly check the location and/or availability of another wireless customer is one of the most promising new features of wireless data networks. Yet it comes with an equally vexing problem: how to control the privacy and security of this data. Users must be confident they have complete authorization and access control over sharing of PAM data or they may turn this feature off altogether.

Therefore an interoperable trust model may emerge as the pre-eminent requirement for calendar sharing and PAM. However the payoff in personal and business productivity if this hurdle can be overcome is enormous. The labor savings for administrators and participants in countless daily calls, meetings, conferences, and other events would be on the order of the labor savings that Internet email brought to message delivery or the Web brought to information retrieval.

[2] http://www.oasis-open.org/committees/ciq/
[3] http://www.imc.org/pdi/
[4] http://www.slipstick.com/outlook/share.htm
[5] http://www.parlay.org/about/pam/index.asp

### *Trusted Search*

Internet search services like Google, Yahoo, and MSN are tremendous utilities for finding information. However they all suffer one universal limitation—they are limited to public data, the data that can be retrieved by automated robots from the public Web. Although these same technologies can be implemented within private networks to build private search indexes that are only searchable by authenticated users of each network, this effectively divides the universe of information into one very large public library and many very small private libraries and nothing in between.

This is not, of course, how access to protected information works in the real world. Some of the most valuable information comes through trusted data sharing networks that span multiple personal or business communities. Asking your network of friends for a local car mechanic they recommend, or contacting your school for the phone number of an old classmate, or searching a network of business partners for contacts to a prospective customer, are well-established patterns of trusted information sharing.

Precisely because such information is so valuable, some distributed communities of interest are now seeking to apply Web search technologies across their networks. Global organizations such as the Impact Alliance[6], the Interfaith Community Network[7], and the international law enforcement community[8] want to enable searching for high-value non-public information across the network of individuals, groups, or enterprises with which they have the necessary trust relationships. In doing so they face major obstacles with currently available technologies. Specifically:

- How can the identity of a user in one community be shared across the other communities in the network?
- How can users control the availability of their personal profile or relationship data for searches by other communities, or by specific subgroups of a community?
- How can trust relationships between members of a community, or between communities, be formed quickly and easily?
- How can data sharing permissions be managed dynamically by all members of the network?
- How can new and updated information propagate to all interested parties without being mistaken for spam?

Again these issues illustrate that distributed data sharing is much more than just a technology problem. The fourth dimension of this challenge—the need for an interoperable trust model—is primarily an issue of being able to quickly and easily map real-world trust relationships onto the network.

[6] http://www.impactalliance.org/
[7] http://www.nain.org/info/directory.htm
[8] http://lists.oasis-open.org/archives/legalxml-courtfiling/200309/doc00000.doc

# How the Dataweb Provides a Generalized Solution

None of the distributed data sharing problems described above are new; they are simply growing more apparent with the advent of Web services. Yet Web services alone does not appear to offer a solution. For example, even if the members of a major standards body could agree on Web service definitions for a dynamic address book service, a calendar sharing service, or a trusted search service, how would it be adopted? Would every service provider need to implement each of these Web services separately? How would each of these Web services share data between each other? What would happen if there were competing Web service definitions from different standards bodies?

These questions highlight the fact that distributed data sharing is a *generalized* problem. Certain types of data (e.g., contact data, profile data, ontology data) need be shared almost universally across domains and applications. In this context, one of the core tenets of Tim Berners-Lee's original design for the Web stands out: it was not specific to any particular type of content, application, or usage domain. Rather it was a simple, generalized model for sharing and linking documents over the Internet that could be used for *any* type of content by *any* type of author in *any* information-sharing context.

In short, the Web was the “simplicity on the other side of complexity” for cross-domain sharing and linking of content. It solved the general problem by providing three simple standards for:

- Globally identifying documents (URIs).
- Representing and linking documents (HTML).
- Exchanging documents (HTTP).

The Dataweb applies this same approach to distributed data sharing. The Dataweb model proposes to solve the general problem by providing three simple standards for:

- Globally identifying data and data authorities using *XRIs* (Extensible Resource Identifiers), a new URI-compatible scheme developed by the OASIS XRI TC for abstract identifiers that are independent of a specific location, application, directory, or domain.
- Representing and linking data using the *XDI meta-schema*, a simple XML schema that uses XRIs to identify, describe, and link distributed data in a domain-independent format called an *XDI document*.
- Exchanging data using *XDI service*, a generalized service for sharing and synchronizing XDI documents with bindings to HTTP, SOAP, SMTP, and other transport protocols.

There is also a fourth element to Dataweb architecture does not have a direct corollary in today’s Web. This is the element that addresses the most challenging “fourth dimension” of cross-domain data sharing described above: human trust relationships. This element is called *XDI link contracts*, and to understand them, we must delve a little deeper into the structure of the Dataweb.

### *Dataweb Links: Pipes vs. Strings*

Web and Dataweb architectures are so similar that the primary difference lies in one feature: while Web links are one-way "strings" between HTML resource representations, Dataweb links are two-way "pipes" between XML resource representations. This is illustrated graphically in Figure 1:

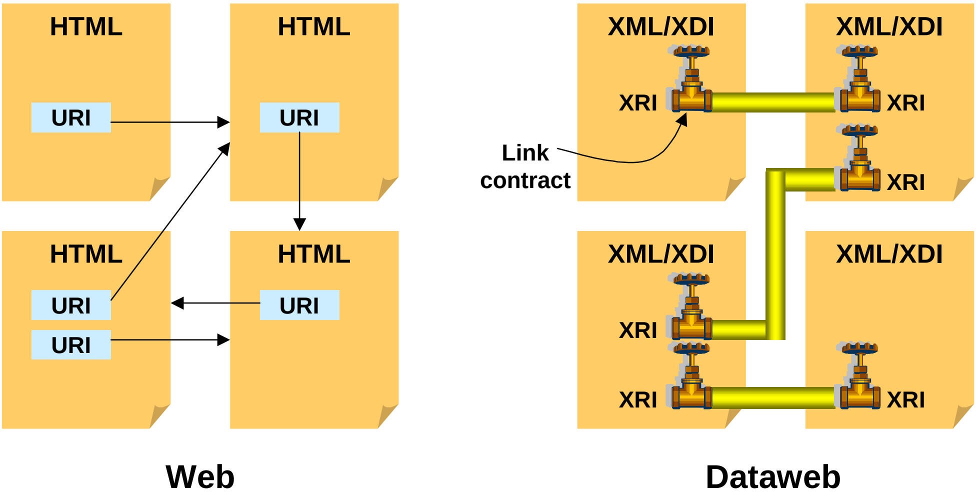

*Figure 1: Web "strings" vs. Dataweb "pipes".*

How exactly is a Dataweb pipe different from a Web string?

1. *Two-way data flow.* A Web string (URI link) can only "pull" linked resources. By contrast, a Dataweb pipe can both pull and push data in either direction.
2. *Persistence.* A Web string is either working (it retrieves a resource) or broken (it produces an error). By contrast, Dataweb pipes can use persistent XRIs that allow the link to reestablish itself even if the target resource moves, changes its name, or changes owners.
3. *Data control.* Access to a Web resource is either public (uncontrolled) or private (credential-protected). By contrast every Dataweb pipe can have a "valve" on each end—an XDI link contract—that allows comprehensive, fine-grained control over all data flowing through the pipe.[9]

## *Dataweb Link Contracts: Data Flow Control Valves*

This last feature of Dataweb links is critical to building Dataweb trust relationships. On the Web, links are "one-dimensional", i.e., they use the HTML "A" (anchor) tag to specify a single URI for a target resource. On the Dataweb, links can be full three-dimensional resources, i.e., active XDI documents that reference other XDI documents just like a real-world contract might reference items of real property.

This three-dimensional approach to Dataweb links can provide:

1. *Active identification.* A Dataweb link can contain any number of XRIs for the target resource (different XRI *synonyms* can be used for different identification, resolution, and data protection purposes—see below). And these XRIs can change as the target resource's context evolves.
2. *Active data interchange control.* A Dataweb link contract can describe the controls on all facets of the data interchange relationship between the source and target resources (see below.)
3. *Active data caching*. Due to the simple structure of the XDI meta-schema, a Dataweb link can optionally cache linked data just by "cutting-and-pasting" XDI documents.

What types of controls can XDI link contracts define? Anything necessary to mediate a data interchange relationship. They are as flexible and extensible as real world contracts (e.g., non-disclosure agreements). For example, link contracts can mediate:

[9] Note that XDI link contracts are not required on every Dataweb link—they are only needed when it is necessary to control the flow or caching of shared data.

- *Authentication.* How are communications with the target resource authenticated? If XML digital signatures are to be used, where is the public key certificate?
- *Authorization.* What privileges to which resources is the target resource granted?
- *Access control.* What access permissions are granted to what data?
- *Usage control.* For what purposes can this data be used?
- *Distribution and forwarding control.* To whom can this data be redistributed and for what purposes?
- *Synchronization.* How will the link cache be updated if data changes in the target resource?

Furthermore, since link contracts are themselves XDI documents, they can (like real-world contracts) govern their own revision, amendment, and deletion.

In sum, Dataweb link contracts permit fully distributed peer-to-peer data mediation the same way Web links permits fully distributed peer-to-peer content access. They are a solution that models (and scales to) the real world of data sharing.

## *Dataweb Addressing: XRIs and XRI Synonyms*

After link contracts, the second most important contrast between the Web and the Dataweb is addressing. Both architectures have the same fundamental premise: 100% addressability of every resource. However on the Dataweb this applies to *every single item of data*, down to the smallest individual attribute.

This is an example of how the addressing requirements for Internet-scale data sharing are more stringent than for content sharing. These requirements are covered in detail in *XRI Requirements and Glossary 1.0*,[10] published by the OASIS XRI TC.[11] A brief summary is provided in Table 1:

| Requirement | Web (HTTP URIs) | Dataweb Addressing (XRIs) |
|---|---|---|
| Persistence | No standard solution for addresses that do not break when a resource moves or changes names (the Web equivalent of a database global foreign key). | Must support both persistent and reassignable identifiers at any level. |
| Unlimited delegation & federation | Supports delegation only in DNS names (e.g., the first segment of an HTTP URI —"www.example.com"). | Must support delegation of both persistent and non-persistent identifiers to any depth. |
| Global contexts | DNS top-level domains provide limited global context for an identifier. (Exceptions are restricted TLDs like ".name" for individuals and ".pro" for professionals.) | Must support four global contexts – personal, organizational, general, and special. |
| Cross-references | URIs cannot be nested in other URIs to permit sharing of identifiers across domains. | Must allow XRIs (and URIs) to be nested within XRIs to any depth. |
| Self-references | No syntax for referring to "the concept represented by an identifier" rather than an actual resource to which the identifier resolves. | Must provide standard syntax for self-references. |
| Internationalization | Partially internationalized. IRIs (Internationalized Resource Identifiers) in progress. | Must be fully IRI compliant. |
| Extensibility | Extensible through new URI schemes and resolution protocols. | Must be extensible within the XRI scheme and resolution protocol. |

*Table 1: The requirements for Dataweb addressing.*

[10] http://xml.coverpages.org/XRI-RequirementsV10-20030612.pdf
[11] http://www.oasis-open.org/committees/xri/

In addition to the new capabilities of XRI syntax, XDI also supports *XRI synonyms*. An XRI synonym is established whenever two or more XRIs—regardless of their syntactic equivalence—are asserted in an XDI document to represent the same resource. XRI synonyms are the way XDI can maintain both human-friendly reassignable XRIs (*e-names*) and machine-friendly persistent XRIs (*e-numbers*) for the same resource at any scope—global, local, or relative. So a resource can have both multiple e-names (e.g., in different languages) and multiple e-numbers (e.g., for different data sharing and privacy contexts)—all managed automatically by how the XRIs are assigned and shared using XDI link contracts.

In short, XRI addressing is how XDI can solve one of the most challenging problems in enterprise application integration (EAI): mapping and translating data between domains. With XDI you can finally, "Map once, link everywhere".

## *Dataweb Representation: The XDI Meta-Schema*

The breakthrough of HTML was that it provided a simple, application-, domain-, and language-independent format for content representation. Without it, the Web could never have achieved ubiquity. To emulate this vital feature, the Dataweb must do the same for data representation in XML.

On the surface this would appear to be an immense challenge due to the enormous variety of existing XML schemas and schema languages. However the purpose of the XDI meta-schema is quite the opposite: to provide a simple, universal, “schema-neutral” format for exchanging, linking, and synchronizing other XML-encoded data (or references to other URI-addressable data). In essence, XDI is an XML "wrapper" or “descriptor” for instances of XML defined in any schema language.

Thus the proposed XDI meta-schema (see Appendix A) has only four XML elements (and no XML attributes) as shown in the UML relationship diagram in Figure 2:

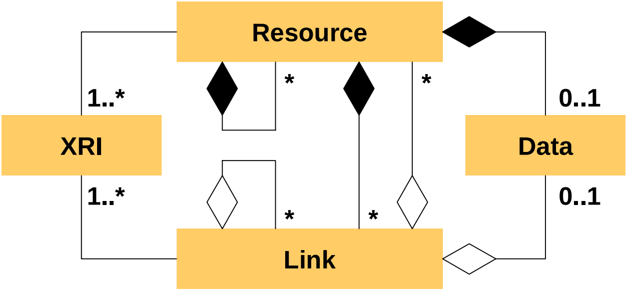

*Figure 2: UML model of the XDI meta-schema.*

- *Resource* elements represent any XRI-identified resource. They must contain at least one XRI, but can contain any number of additional XRI synonyms. They contain only one data element, but they can contain any number of link elements, and can recursively contain other resource elements. As containers, resource elements express UML composition relationships, i.e., “ownership” of the contained resource.
- *Link* elements are "mirrors" of resource elements except they express UML aggregation relationships, i.e., “membership” rather than “ownership”. Like resources, a link must always have at least one XRI associating the resource it represents, but it can have any number of additional XRI synonyms, all of which also identify the linked resource. This is the key to building dynamic XDI data maps.
- *XRI* elements represent UML association relationships. They can contain only valid XRI values (which include all valid URIs, since XRIs are a superset of URIs).
- *Data* elements are the "primitives" of XDI—the edge nodes of the XDI graph. They carry the actual data payload—anything from a literal XML character string to an entire XML document. The data element uses the XML Schema "Any" content model so it can contain any well-formed XML.

Like HTML, the purpose of the XDI schema is to reach "the simplicity on the other side of complexity" for distributed data sharing. Despite its simple structure, the expressive power of XRIs allows instances of the XDI meta-schema to identify, exchange, link, and synchronize any XML-encoded or URI-referenceable data between any two locations or applications on the Internet.

### Dataweb Services: The XDI Service Definition

Of all the components of Dataweb architecture, the one that most resembles its Web ancestry is the proposed XDI service definition. Although defined abstractly using WSDL, it is proposed to consist of the same four basic operations as HTTP:[12]

- *GET* is the same universal workhorse as in HTTP, i.e., it retrieves an XDI document using an XRI.
- *SET* is the equivalent of HTTP PUT, and is used to write XDI documents.
- *DELETE* is identical to HTTP and is used to delete XDI documents.
- *POST* serves the function of invoking any XRI-identified operation upon the posted XDI document and optionally receiving an XDI document in response.

The advantage of defining XDI service abstractly in WSDL is that it can be bound to any transport protocol. So even though XDI can be used "natively" over HTTP, it is equally at home using the full XML enveloping and messaging capabilities of SOAP. XDI can also be bound to SMTP/MIME as a simple way to extend structured data interchange to Internet email.

## Applying the Dataweb Model

Having described the Dataweb model, now let’s apply it to our three distributed data sharing problems.

### Business Card Sharing Using the Dataweb

The first challenge is to share and link electronic business cards between dynamic address books—and do so in a way that is interoperable across domains like Internet email and the Web. As explained above, the Dataweb approach to this problem is just like the Web. Specifically, once Web servers and browsers were available, all an author had to do to begin sharing information over the Web was:

1. Obtain a Web address (if the author didn’t already have one);
2. Put up a Web site with the content (HTML pages) the author desires to publish; and
3. Start publishing the Web addresses (URIs) of these pages.

To begin sharing and linking data such as a business card over the Dataweb, the process is nearly identical, with the addition of one more step:

1. Obtain an Dataweb address (if the author doesn’t already have one[13]);
2. Put up a Dataweb site with the data (XDI documents) the author wants to share;
3. Create the Dataweb link contracts (other XDI documents) describing the terms under which this data may be shared; and
4. Start publishing the Dataweb addresses (XRIs) of these link contracts.

[12] The OASIS XDI TC will also consider whether XDI should include protocol-level support for basic transactional integrity.

[13] Remember that XRIs are backwards-compatible with most HTTP URIs, so most Web authors can begin using the Dataweb immediately

Note these last two steps: *on the Dataweb, protected resources are never addressed directly*. They are always addressed through the link contract that establishes the data sharing relationship between the publisher and any subscriber. (This is sometimes called the "rights path".)

Let's illustrate this in the context of a business card sharing relationship. Say Alice and Bob want to begin sharing electronic business cards, and both of them have Dataweb addresses (XRIs for their Dataweb home pages) from an XDI service provider. From a Dataweb perspective an electronic business card is simply an XDI document containing instances of contact data.[14] And a dynamic address book is simply a Dataweb site (or a branch of the site) devoted to sharing and linking XDI business cards.

Alice and Bob can each use a Dataweb editor[15] to create an XDI business card containing the contact data they would like to share. This is published to their respective Dataweb sites as shown in Figure 3.

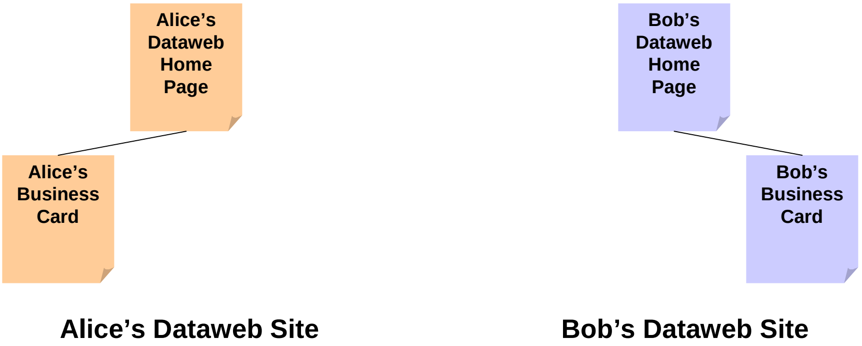

*Figure 3: Two Dataweb sites containing electronic business cards as XDI documents.*

Alice must next choose a "template" link contract for sharing business cards. Typically this will be similar to a real-world contract template—a standard form with a handful of customization options.[16] For example, it may let Alice decide the default permission is to permit a contact to forward her XDI business card directly to others ("introduction permission"), vs. needing to be asked first. In all cases, when Alice is finished customizing this template, she publishes it to her Dataweb site as shown in Figure 4.

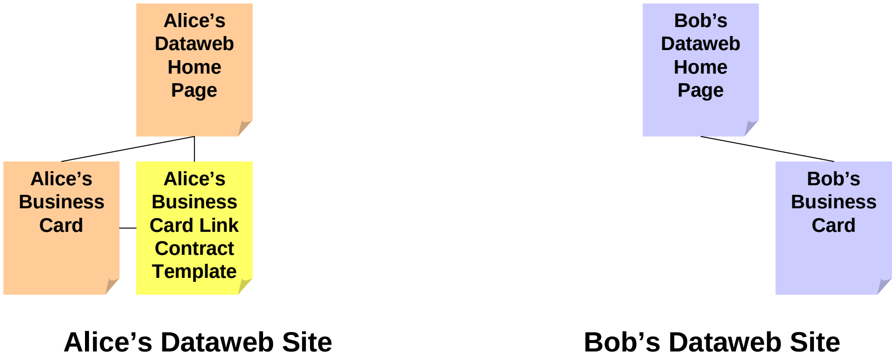

*Figure 4: Alice adds a link contract for sharing her business card.*

[14] Most likely these datatypes will be identified with general XRIs assigned to standard contact datatypes such as those developed by the OASIS CIQ TC. See also the discussion of general XRIs and XDI dictionaries below.

[15] This may be an online HTML application provided by an XDI service provider, or a new feature provided by a client-side contact manager such as Microsoft Outlook or Palm Address Book.

[16] XDI personal link contracts in particular are expected to be very simple, reflecting the direct nature of most person-to-person trust relationships.

To share this business card with Bob, Alice sends him the XRI for the link contract. This is where XDI service comes into play. When Bob clicks this link, Alice's XDI service provider redirects him to his own XDI service provider. After authenticating Bob, Bob's service provider makes an XDI request to Alice's Dataweb service provider to retrieve the link contract using the XRI. Providing Bob agrees to Alice's contract, Bob's server returns a signed version of the contract[17] and keeps a copy for Bob. Finally Alice's server returns a copy of Alice's business card to be cached by Bob's site, as shown in Figure 5.

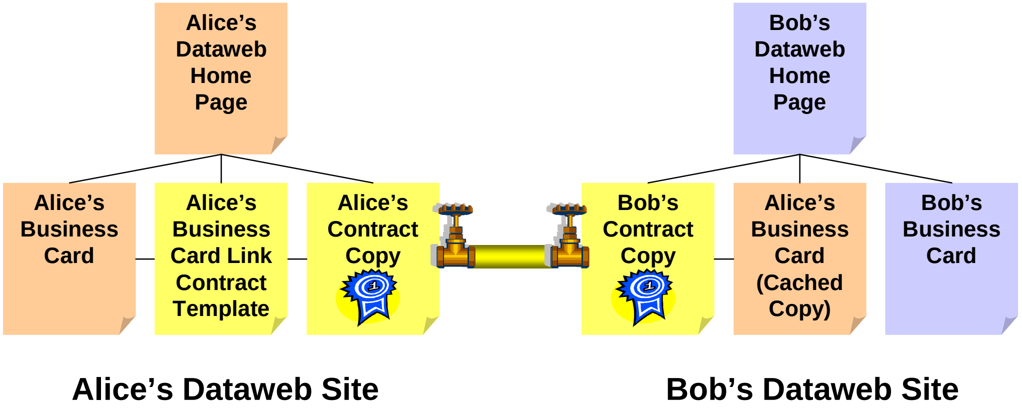

*Figure 5: Both sites have a copy of the signed contract and Bob's site has a cached copy of Alice's business card.*

Providing Alice has a persistent XRI, Alice and Bob have just created a lifetime link—one that won't break no matter how many times Alice moves, changes jobs, or even changes XDI service providers. And if Alice's contract includes data synchronization permission, updates to Alice's contact data will automatically flow across to Bob's site (via an XDI update message sent from Alice's XDI service provider to Bob's XDI service provider).

Bob can share his XDI business card with Alice the same way. This mirrors the exchange of XDI documents shown in Figure 5, and sets up a second "pipe" between their two Dataweb sites to keep Bob's business card synchronized with Alice's cache.

The advantage of this model for business card sharing is that, like the Web, it is the same for *all* Dataweb sites—person-to-person, person-to-business, business-to-business. The types of contact data and the terms of the link contract may vary, however for most common types of data sharing transactions (like business cards), the link contract will in all probability be very simple and standardized.

This model extends within a user or organization's *internal* Dataweb as well. For instance, rather than having to look up Alice's phone number online at his XDI service provider, where Bob really needs this data is directly on his laptop and cell phone. If both include XDI dynamic address books linked to his online Dataweb site, Alice's business card and any updates would automatically flow through his XDI service provider right down to his local devices. In essence, these devices become local caches of a selected portion of Bob's online Dataweb site.

## *Calendar Sharing Using the Dataweb*

We've seen how Dataweb links are generated between Dataweb sites to set up persistent data sharing relationships. Now how is this applied calendar sharing?

[17] This could be using an XML digital signature, or whatever other means of verification is acceptable to Alice.

An appointment is just another group of shared data, so from a Dataweb standpoint every appointment is simply a new XDI document that needs to be shared among all attendees. However, following the Dataweb rule of access, this first requires that attendees have a contract covering this type of data. Such a contract may include options such as, "Can this contact automatically schedule an appointment on my calendar? Can they move an appointment? Can they see an appointment that they have not created?"

Many XDI business card sharing contracts are also likely to include basic calendaring sharing permissions. Let's say this is the case with Alice's contract with Bob, and now Bob wants to schedule a meeting with Alice. Bob would use his Dataweb editor to create a new XDI appointment and "invite" Alice by linking it to her contract, as shown in Figure 6.

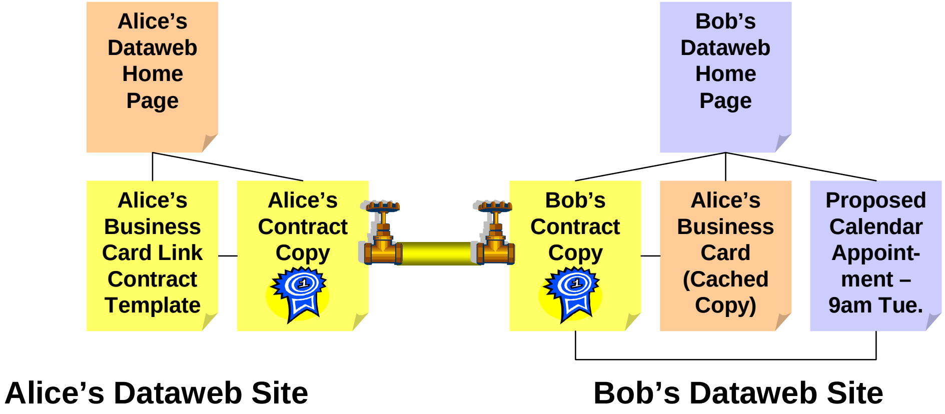
Alice's Dataweb Home Page
Bob's Dataweb Home Page
Alice's Business Card Link Contract Template
Alice's Contract Copy
Bob's Contract Copy
Alice's Business Card (Cached Copy)
Proposed Calendar Appoint-ment – 9am Tue.
Alice's Dataweb Site
Bob's Dataweb Site

*Figure 6: Bob creates a proposed appointment linked to Alice.*

Bob's XDI service provider sends the XDI appointment document to Alice's XDI service provider. Assuming Alice handles appointment requests locally, Alice's service provider would deliver the message to her preferred device (such as her laptop or cell phone), which would notify her as she desires (such as using an HTML email or SMS message). Alice can now accept, decline, or modify the request as with most calendar sharing systems. If she accepted it, the result would look like Figure 7 (except that this appointment would also be cached locally on her laptop or cell phone, not shown.)

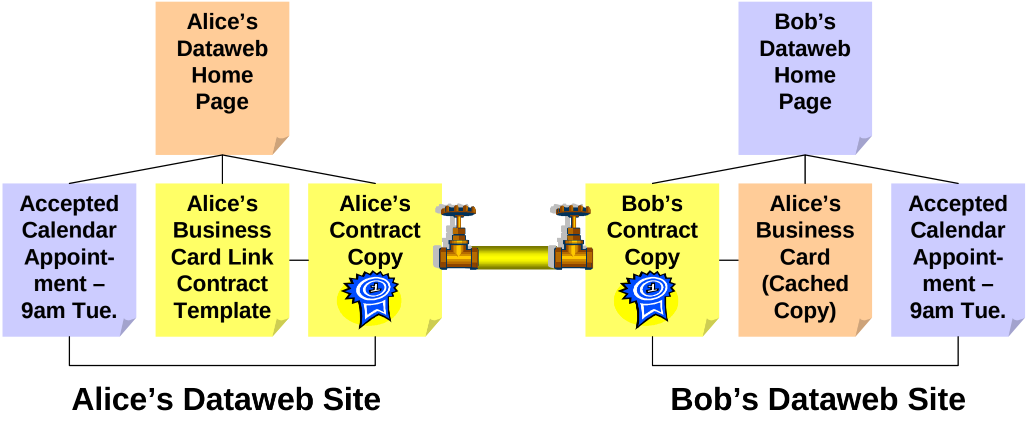
Alice's Dataweb Home Page
Bob's Dataweb Home Page
Accepted Calendar Appoint-ment – 9am Tue.
Alice's Business Card Link Contract Template
Alice's Contract Copy
Bob's Contract Copy
Alice's Business Card (Cached Copy)
Accepted Calendar Appoint-ment – 9am Tue.
Alice's Dataweb Site
Bob's Dataweb Site

*Figure 7: Alice accepts the appointment and it is confirmed on Bob's site.*

The real productivity payoff is realized when conflicts arise and either Bob or Alice needs to move the appointment. Assuming both have permission, either could modify it, and the other will receive the proposed change automatically. Negotiation over conflicts and proposing alternative times can also be automated to the extent that Bob and Alice's respective XDI calendaring applications support this.

None of this is new functionality—these features are staples of workgroup calendaring systems. The breakthrough is reaching a platform- and domain-independent solution that can make Internet calendar sharing as widely available as Internet email and the Web.

### *Trusted Search Using the Dataweb*

The third distributed data sharing challenge was being able to do searches across private sites—data that is not available publicly, but that could be very valuable if searchable across a federated trust community.

Internet search engines today compile their indexes using automated "spiders", programs that crawl Web sites and follow the links to discover new content.[18] Once more, the Dataweb approach mirrors the Web approach with the same twist: rather than just crawling links, Dataweb spiders will crawl *link contracts*.

In other words, on the Dataweb, data is only indexed by permission, and that permission is provided to search spiders the same way it is provided to any other party, using link contracts. In turn, Dataweb search engines will provide access control to these special indexes by authenticating search requests with the governing community.

To illustrate, let's say Alice and Bob are both members of a scientific association that offers the benefit of being able to do searches across portions of member's private profile data (for example, their plans to do future research in a specific area.) Offering this feature today would require building a special search site to index the data, creating user accounts on this site for all participating members, and requiring members to login to this site to perform trusted searches.

The Dataweb solution requires only that all participating members share a common link contract. This contract would not only govern what data was being made searchable under what terms, but also provide each member with permission to conduct searches across the data linked to this contract. No special sites, no separate accounts, no additional passwords. Furthermore the system can grow and evolve simply by making changes to this common contract, for example, to add new types of data.

Whatsmore, trust communities can be now be federated by *linking their contracts*. So if Alice's and Bob's scientific association were to form an alliance with another association, and the two communities wanted to offer trusted search across their combined membership, this could be accomplished simply by creating a new contract that links their respective member search contracts.[19] Now a trusted search request from a member of either association would automatically be authenticated against both search indexes.

[18] Google also pioneered the use of link metadata to optimize the resulting indexes.

[19] This contract could also establish the XRI synonym mapping if the two communities were using different XDI datatypes.

# How the Dataweb Builds on Web Services

Although the data model for XDI is rooted in Web architecture, and its WSDL definition can be bound to other transport protocols besides SOAP, XDI is ideal as a Web service because it adheres to the classic tenets of Web services architecture: domain-independent application-to-application XML messaging using XML schema definitions, WSDL service definitions, and SOAP envelopes.

Thus XDI/SOAP is fully compatible with all emerging Web services standards including WS-Security, WS-Federation, WS-Inspection, WS-Routing, and WS-Reliable Messaging. So where does XDI fit with other Web services?

## *A Generalized Data Sharing Service*

In a world of domain-specific Web services, the role of XDI is *as a utility service*—a generalized Web service employed by other Web services to handle common data sharing transactions. For example, say a heavy equipment manufacturer offers a suite of Web services for generating orders, accepting invoices, and monitoring inventory from its supply chain. This suite uses a set of XML schemas and WSDL service definitions customized for the manufacturer's product line and accounting systems.

If this manufacturer wanted to add a new supplier and its sales representatives to its supply chain, the manufacturer could offer another special Web service for accepting this supplier's account data into the manufacturer's CRM. Alternatively the CRM vendor could offer a pre-packaged Web service for doing this same task. But either case the supplier would have to implement another Web service client for communicating this account provisioning data. If the supplier serves 25 manufacturers, this could mean implementing 25 different account-provisioning clients all for the same standard data.

This would become much simpler and less costly if all parties to this supply chain used a general data sharing service like XDI. Then one set of XDI transactions could transfer, link, and permanently synchronize all the account data. XDI could also be used to "bootstrap" the other specialized Web services by transferring the XSD and WSDL files (encapsulated as XDI documents) necessary to configure the supplier's Web service clients. Now the manufacturer's and supplier's respective systems would be ready to begin automated data interchange at a fraction of the time and expense.

## *A Distributed Data Mediation Service*

A standardized data interchange service not only simplifies the creation of a new relationship, it also automates many standard maintenance tasks during the lifetime of the relationship. For example, if the manufacturer and supplier described above used XDI to link their shared data, then changes in either of their configurations—prices, contracts, sales representatives, XSD files, WSDL files—could flow automatically from one's XDI servers to the other's using the mediation of XDI link contracts without any custom programming or specialized administration.

XDI is also a very efficient way of applying SAML authentication and authorization assertions, because in XDI every policy decision point is identified by the XRI for the authoritative link contract. And shared access control policies described in a language like XACML could be implemented within or across organizations—even throughout the manufacturer's entire supply chain—by using shared XRIs to the same XDI link contract.

Lastly, this is a context in which persistent XRIs—XRIs that will not change for the lifetime of a relationship—provide significant benefits. For example, suppose the supplier underwent a major reorganization of its sales force, or was bought by another vendor and changed its name. The use of persistent XRIs means all the XDI links and contracts between the supplier and manufacturer will remain intact even when the DNS names changed. Furthermore, XDI synchronization would allow changes to attributes such as titles, roles, and contact data to automatically flow between the XDI systems.

### *XDI Forms*

The scenarios above raise the question of how XDI is customized for specific types of data interchange. For example, different XDI data sets are needed to provision the supplier's accounts for a heavy equipment manufacturer vs. an airplane manufacturer vs. a plastics manufacturer. How does each manufacturer request just the specific data needed by their CRM system from the supplier's XDI service?

The Dataweb answer is identical to the Web answer: forms. Just as a Web server can request customized data from a Web browser using an HTML form, an XDI client or server can request instances of specific XDI datatypes from another XDI service using an XDI form. The key difference between HTML and XDI forms is that the latter are XDI documents designed for fully automated machine processing. In other words, an XDI form uses machine-readable:

- *Datatypes.* Instead of human-readable field labels, XDI forms use XRIs to identify the XDI datatype instances being requested (including XRI synonyms for domain-independent type mapping).
- *Link contracts.* XDI forms can specify one or more XRIs for the proposed link contract(s) governing the requested data.
- *Credentials.* The link contracts can also include one or more XRI-addressable authentication credentials, which themselves include XRIs for the issuing XDI authorities for verification.

XDI forms can also be chained together to provide workflow and multi-party data sharing transactions. In short, they are the mechanism that enables XDI to serve as a generalized solution to sharing many different types of data under many different contract terms and trust models.

## How the Dataweb Builds on the Semantic Web

The foundation of the Semantic Web—the W3C's term for a Web of machine-readable and understandable resources as opposed to human-readable content—is RDF (Resource Description Framework). RDF is an abstract model and XML syntax for expressing metadata about any URI-identified resource, from a Web page to a file to a media stream. The goal of RDF is to provide a common format for machine-understandable metadata that can be processed by any RDF-aware application.

Just as XDI can integrate directly with other Web services because it can be a Web service itself, XDI can integrate directly with RDF because:

- Every XDI resource is URI-addressable,[20] so it can be described using RDF.
- Every RDF graph can be serialized in RDF/XML, so the resulting XML document is a resource that can be identified, exchanged, linked, and synchronized using XDI.

However, since both XDI and RDF represent graphs of relationships between resources, they can interoperate at an even deeper level.

### *RDF Graphs vs. XDI Graphs*

RDF represents all relationships using tuples—three part statements consisting of a subject, predicate, and object.[21] For example: a book (subject) has a rating (predicate) of "four stars" (object). The descriptive

[20] Every XDI resource is XRI-addressable, and every XRI can be transformed into valid URI.

power of RDF derives from the fact that any value of any property of any resource can be described using this basic construct.

RDF statements can also be nested, i.e., one statement can describe another. For example, you can have an RDF statement describing *who said* that a book has a rating of four stars. But since RDF syntax is based on tuples, doing this actually requires four tuples (a tuple for the new statement, and three tuples describing the new statement's relationship to the subject, predicate, and object of the first statement.) Known as reification, this is one of the most complex topics in RDF.[22]

Since distributed data sharing depends heavily on describing nested data relationships, the XDI meta-schema takes a slightly different approach. It models data from the "inside out", i.e., by starting with a Data element for containing atomic data values and then using three elements to model the three types of UML relationships in which a Data element can participate:

- *XRI elements* model association relationships. The value of the XRI identifies the associated resource.
- *Resource elements* model composition relationships. Resources can be nested to any depth.
- *Link elements* model aggregation relationships. Links can be nested to any depth.

The result is a model for graphing data relationships that is optimized for generalized distributed data sharing. The key word is *optimized*. In theory, both RDF and XDI graphs can describe the relationships between any set of URI-identifiable resources. But each graph is optimized for different purposes. The distinction is similar to that between Cartesian and radian coordinates—either can locate any point on a plane, but certain types of shapes (e.g., squares vs. circles) are much easier to describe using one or the other.

## *The XDI Graph: Optimized for Resource Linking*

In particular, the XDI graph is optimized for describing relationships between shared resources. This is the nature of distributed data sharing. Every time data is shared, a copy of a resource—a new sub-tree—is instantiated at a new location on the global XDI graph. Because this new sub-tree has an inherent relationship to the original sub-tree, it can be represented simply by using:

- A Link element (rather than a Resource element) to represent the root of the new sub-tree, and
- One or more of the same XRIs that described the original Resource element, now in a new context.[23]
- A cache of any data from the original sub-tree that is literally "copied-and-pasted" below the root Link element in the new sub-tree.

The result is both a simple, efficient XDI operation and clean, relatively compact XML.

A second way the XDI graph is optimized for linking is the use of XRIs to identify every resource and link element. Because of this, every node in the XDI graph is directly addressable using both persistent and reassignable XRIs. In addition every node that has a global context (i.e., a globally addressable ancestor) is globally addressable.

A third way the XDI graph is optimized for linking is link contracts, discussed above. Every link element that needs to control its active data interchange relationship with the target resource can cache a copy of

[21] From an object-oriented perspective, this is the same thing as an object, property, and property value. See http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/#basic.

[22] See http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/#higherorder.

[23] Note that for privacy and data protection purposes, the XRIs at any Link node may be only a subset of the XRI synonyms used at the target Resource node (or at other Link nodes) for this resource. This is a secondary mechanism by which data authorities can control the sharing of their data in addition to link contracts.

its link contract, as shown in Figure 2 above. This means each XDI link can be not just self-describing, but self-governing—an ideal architecture for distributed data mediation.

### General XRIs and XDI Dictionaries: Concept Mapping on the XDI Graph

There is a fourth way in which the XDI graph is particularly optimized for distributed data sharing. Since data describing the same resource is often already widely distributed (such as the same registration data representing the same user on multiple Web sites), a key challenge in establishing a distributed data sharing infrastructure is "linking it up". This requires:

1. Establishing equivalence between the different instances.
2. Establishing chain-of-authority relationships (in XDI graph terms, this means deciding which instances are resource nodes vs. which are link nodes).
3. Adding any necessary link control metadata, i.e., link contracts.

The latter two problems hinge on solving the first one—mapping equivalence. XDI offers four tools specifically to help with this problem:

- *XRI synonyms*, discussed above, allow multiple XRIs to be assigned to the same resource or link element. This is how the XDI graph forms a dynamic global "identity map".
- *Persistent XRIs*, also described above, enable XDI authorities to assign persistent IDs to any node in the XDI graph. Most data instances already have a persistent ID within their local data store (e.g., a database key or file system i-node). When adapted to become a persistent XRI, this ID becomes the "anchor point" around which other XRI synonyms can be assigned to build an equivalence map.
- *General XRIs* can be assigned to any resource representing an instance of a shared concept ("phone number", "shoe size", "age", "budget", "whale", etc.) General XRIs use the general context symbol ("+") designed for this purpose. There is no single global authority for general XRIs because XDI communities will self-select a common set of general XRIs the same way communities of same-language speakers naturally self-select a common set of generic nouns. (However general XDI dictionaries will be very useful in this respect.)
- *XDI dictionaries* are Dataweb sites used as references by other Dataweb sites to help build shared concept maps. Except for the fact that they are entirely machine-readable and searchable, their purpose is the same as real-world dictionaries—to share resource identifiers (words) and resource descriptions (definitions) across communities of interest. Like real-world dictionaries, XDI dictionaries can be either generalized (the XDI equivalent of Merriam-Webster’s or the Oxford English Dictionary) or specialized (the XDI equivalent of Black’s Law Dictionary or Stedman’s Medical Dictionary.)

With XDI dictionaries, Dataweb mapping software can help automate the process of assigning XRIs and XRI synonyms to local resources, in the process building equivalence maps for linking this data with other XDI authorities. With such tools it may be possible, despite the four-dimensional challenges of distributed data sharing, to establish a global Dataweb almost as quickly as the World Wide Web was formed.

# The Mission of the OASIS XDI TC

Besides the Web and XML, the roots of the Dataweb are in data addressing and linking technology originally developed by the XNS Public Trust Organization.[24] This work was contributed to the OASIS XRI TC in January 2003 to help develop the abstract, location-independent identifier scheme necessary to support Internet-wide persistent linking and synchronization of data.

With the approval of the XRI Generic Syntax and Resolution Specification 1.0 as a Committee Draft[25] on January 19, 2004, the foundation is now set for the OASIS XDI TC to develop the other specifications required for a global Dataweb. These deliverables include:

- *The XDI meta-schema* for describing and linking XRI-identified resources using XML documents. (See the initial proposal attached as Appendix A.)
- *XRI resolution rules* within XDI documents.
- *The XDI WSDL service definition* and bindings to transport protocols including HTTP, SOAP, and SMTP/MIME.
- *The XDI service dictionary*—the set of globally shared XDI resources including that can be used in XDI link contracts to define, control, secure, and protect data sharing relationships using XDI.

In addition the XDI TC plans to work closely with other OASIS TCs as well as other standards bodies and industry consortiums. Initially the XDI TC will be forming liaisons with:

- *The Trusted Computing Group (TCG)* Infrastructure Working Group to establish common use cases and ensure interoperability of TCG and XDI specifications in trusted computing infrastructure.[26]
- *The Telemanagement Forum (TMF)* to establish common use cases and requirements for using XDI as a mechanism for implementing the TMF SID (Shared Information/Data Model) component of the TMF NGOSS (New Generation Operations System and Software) program.[27]
- *The International Security, Trust, and Privacy Alliance (ISTPA)* Framework Working Group to adapt the ISTPA Privacy Framework as a reference model for XDI privacy and usage control architecture.[28]

We invite other industry alliances, standards bodies, companies, and individuals interested in XDI and Dataweb development to join the OASIS XDI TC or to contact us regarding developing a liaison.

[24] http://www.xns.org
[25] http://www.oasis-open.org/apps/org/workgroup/xri/download.php/5109/xri-syntax-resolution-1.0-cd.pdf
[26] https://www.trustedcomputinggroup.org/
[27] http://www.tmforum.org/
[28] http://www.istpa.org/

# Conclusion

The task of establishing an Internet-wide data sharing infrastructure might appear overwhelming. However ten years ago, the idea of a global Web that put billions of linked documents at our fingertips appeared equally unthinkable. What the Web proved was that a simple, generalized model for content sharing and linking could grow exponentially as each new link was created.

The Dataweb proposes the same model for distributed data sharing. As shown in Figure 8, it combines the best features of the Web's linking model, Web Service's structured XML messaging model, and the Semantic Web's resource description model into an architecture that has the potential do for interoperable trusted data sharing what the Web did for open content sharing.

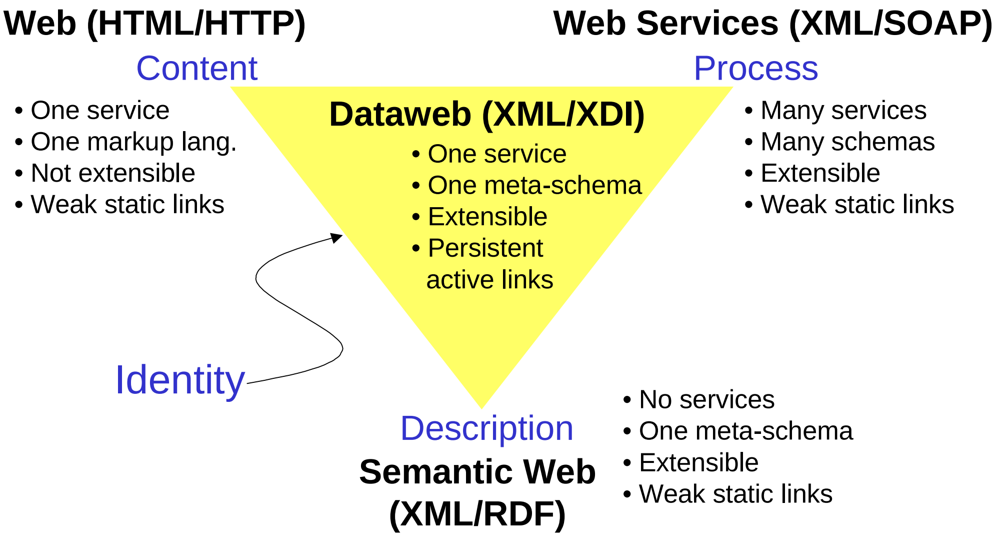

*Figure 8: The Dataweb bridges the best features of the Web, Web Services, and the Semantic Web.*

Moreover, XDI does not displace any of the existing or emerging XML vocabularies designed to support specific applications or Web services infrastructure. Rather, it augments them by providing a common Web service for sharing, linking, and synchronizing other XML documents encoded in any XML language or schema. XDI service is designed to layer over enterprise-level data sharing solutions and standards the same way the Web layered over enterprise-level content sharing solutions and standards.

The organizers of the OASIS XDI TC invite your participation in this work. For more information, please visit the OASIS XDI TC home page at http://www.oasis-open.org/committees/xdi.

# Appendix A: The Proposed XDI Meta-Schema

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://xdi.oasis-open.org"
      xmlns:xdi="http://xdi.oasis-open.org"
      elementFormDefault="qualified">

<element name="data">
  <complexType mixed="true">
   <sequence>
    <any minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>

 <element name="xri">
  <simpleType>
   <restriction base="string">
    <annotation>
     <documentation>
        This will contain an XML pattern element matching the XRI ABNF,
        so a standard XML schema validating parser will be able to
        validate that the XRI value is legal.
     </documentation>
    </annotation>
   </restriction>
  </simpleType>
 </element>

<element name="resource">
  <complexType>
   <sequence>
    <element ref="xdi:xri" minOccurs="1" maxOccurs="unbounded"/>
    <element ref="xdi:data" minOccurs="0" maxOccurs="1"/>
    <element ref="xdi:resource" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="xdi:link" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>

<element name="link">
  <complexType>
   <sequence>
    <element ref="xdi:xri" minOccurs="1" maxOccurs="unbounded"/>
    <element ref="xdi:data" minOccurs="0" maxOccurs="1"/>
    <element ref="xdi:resource" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="xdi:link" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>

</schema>
```