Jamcracker W3C Web Services Workshop Position Paper

Author: David Orchard (Jamcracker) dorchard@jamcracker.com
Date: April 11-12th 2001

Executive Summary

This paper details Jamcracker's position on the directions that the W3C should embark on with respect to various XML, Web and Web Services technologies and processes. Web Services is a concept and technology set that has rapidly grabbed the imagination of users, vendors and customers. At the core of it, Web Services allows for autonomous communications between applications using the same infrastructure that the Web is built on. The lack of mention of a particular technology or process does not imply Jamcracker's support or lack thereof, rather that we choose to highlight certain aspects over others. We expect that many position papers will be submitted on specific solutions for service description, service discovery, security, reliable messaging, transactions, etc. We choose to focus on a few key areas that we encounter as a platform provider.

Background

As Web Services is such a potentially large area to describe and constrain, we will provide some background about Jamcracker and our use of Web services technology. Jamcracker is a platform that provides integration of Application Service Providers, Web services, and customer systems. From our position as the leading ASP Aggregator, we have a unique position on the realities of the Web Services Integration market. Integration occurs at many different levels. There is the popular view of integration, that of a process receiving and returning XML documents, which we call business process integration. Additionally, and also a key need in today's reality, integration of web pages necessitates integration of single sign-on, provisioning, and billing systems. A variety of non-web based systems can be integrated together, such as EAI systems, LDAP, CORBA, etc. Integration of Jamcracker with partner support and help desk systems is required to support all the various modes of integration.

We are one of the first companies that have actually delivered a cross-domain marketplace of web services. We have built a large and growing ecosystem of partners that provide a variety of web services across domains using many integrations in our platform. To build this ecosystem, we have dealt with technology issues involving service description, discovery, security, provisioning, billing; as well as the more complex business issues involving relationships, servicing, service level agreements, and quality of service.

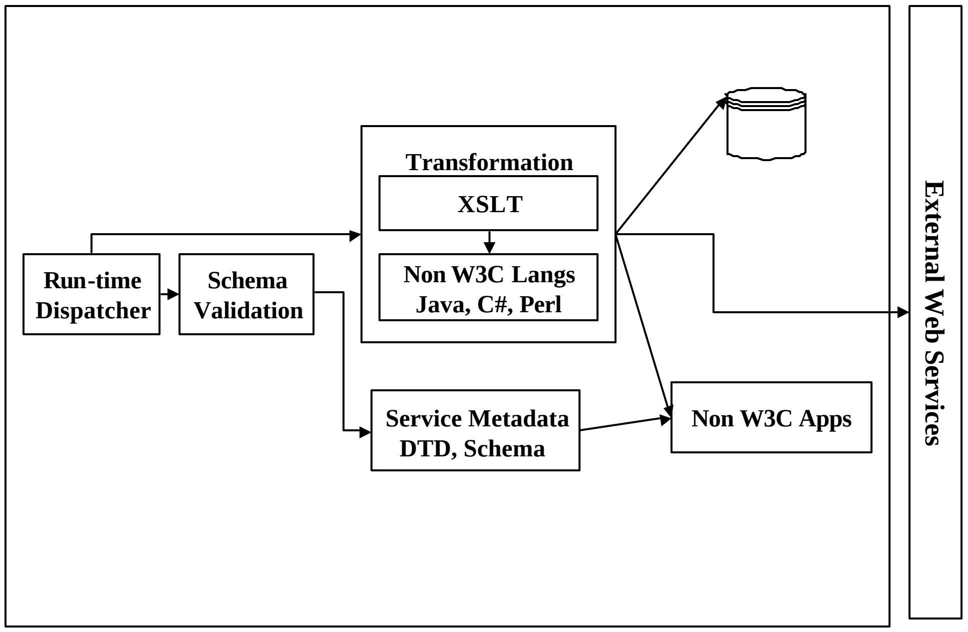
Evolution of XML and Web Services

Our goal in this section is to show a reasonable evolution of the W3C and non-W3C architectures. This allows us to clearly articulate our positions.

Current W3C Architecture

The current web service platform provides a complex workflow of tasks based upon the receipt of a web service request. We introduce a straw man web services architecture. The input is an XML request, encoded in XML Protocol. The runtime dispatcher, typically the first component to receive the request, will execute follow the following steps: 1) Validate request type information. 2) Dispatch to a transformation service. The transformation service can be implemented in a variety of languages, ranging from the W3C's

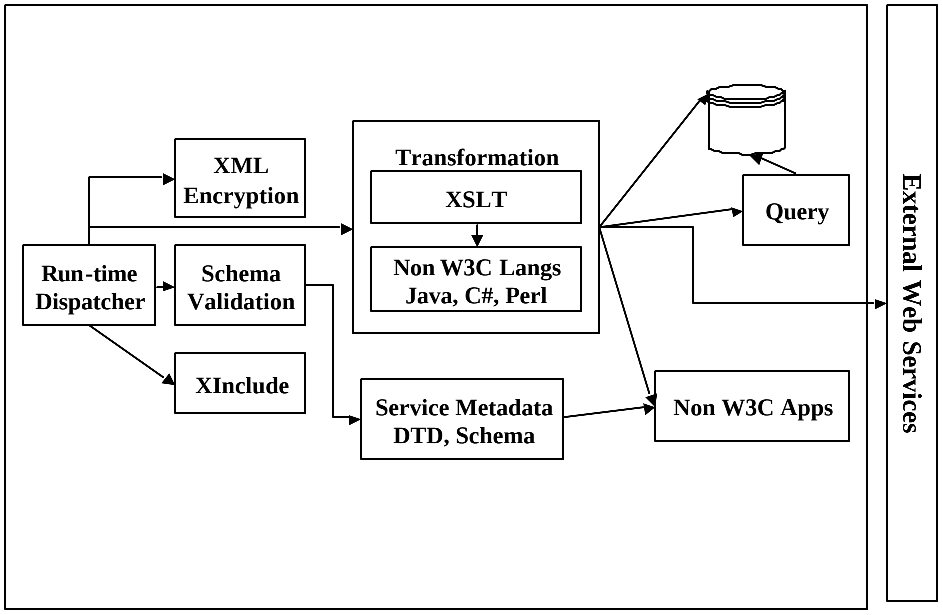
XSLT language to non-W3C languages such as Java, C#, Perl, etc. The transformation language interacts with databases and non-W3C applications. A typical example is a web service that interacts with SAP.



W3C Architecture, XML Next Generation

The W3C has currently chartered various committees to produce XML Encryption, XInclude, and XQuery specifications. Adding these specifications to the architecture, we see that the runtime dispatcher may execute the following steps, in an unspecified order:

- Decrypt request
- Validate request type information
- Perform XInclusions
- Dispatch to a transformation service
- Perform XQueries against documents, databases, etc.



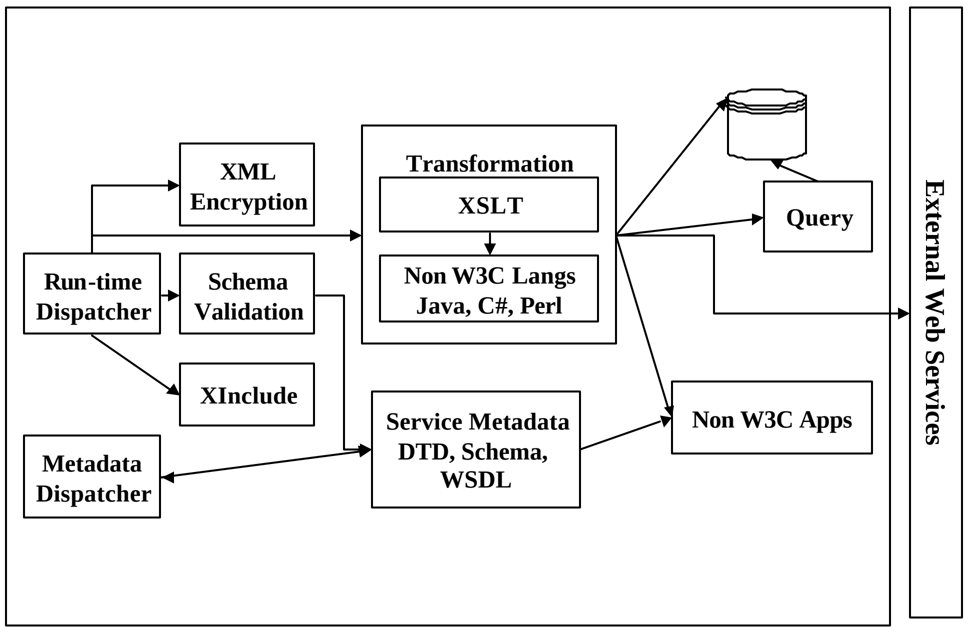
There is a clear and obvious need for defining the various states and processes that the Web service request goes through. This is currently termed the XML Processing Model.

Position #1: Processing Model: The W3C should define the XML Processing model. We do not believe that the processing model should be an area of vendor competition. This is a co-requisite of other web services work. This work may entail refactoring and work in other W3C specifications to ensure uniformity and consistency. Indeed, we have lobbied for years that the W3C should define 1 or more XML "Profiles", ala Sun's Java J2EE ™, that normatively integrates all the XML specifications into a unified architecture. We are impartial to the mechanisms used to achieve an XML Processing Model and any refactoring of XML specification(s).

W3C Web Service Architecture, likely Next Generation

A logical and likely next step is adding Web Service definitions – such as WSDL - and discovery to the Web Services architecture. We now add these to the straw man web service architecture. A new concept is introduced; a metadata request dispatcher. This is a software component that dispatches requests outside the service context. An early form of this is a UDDI repository with it's interface for requests. Typical operations include searching, adding, and modifying metadata entries. It is clear to readers that the metadata dispatcher will probably follow the same processes as the run-time dispatcher – as the metadata dispatcher is a run-time for metadata. It is not relevant to decompose the metadata dispatcher in the same manner as the run-time dispatcher as that would create too much detail in the straw man.

The metadata dispatcher performs queries against all the services that are registered with it. These services can be web services, perhaps defined through WSDL, as well as non-web services. It is crucial that any metadata definition and any repository definition allow for specification of non-W3C resources, ala API calls etc. There is a long history in the web of supporting non-W3C defined resources, specifically the IMG tag. In the same manner that IMGs can be presented to users, non-W3C defined services must be representable in a coherent view of all services available in a platform. A world without web service definitions supporting non-web service interfaces would be similar to a world without IMG tags.



Position #2: The W3C should define a service description language, ala WSDL, probably through a Working Group under the XML Protocol Activity. We expect that Microsoft and IBM will submit a position paper that contains some standardization of WSDL, so we choose not to attempt to duplicate their work.

Position #3: The XPSDL (XML Protocol Service Definition Language, a suggested term for a WSDL WG) WG should have loose charter and not be bound closely to WSDL. We do not wish the same tight coupling to an early specification, such as was done with XMLP and SOAP. Our reasons are that we have not seen sufficient implementations to evidence that WSDL is sufficient and complete, nor has the community at large had a significant opportunity to help WSDL.

Position #4: The charter of the XPSDL WG should allow for service description of non-XMLP applications. We strongly endorse the requirement that WSDL meets in this matter. A large majority of our integrations involve non-XMLP applications, and we have grown tired of learning, teaching and implementing a variety of different interface definition languages.

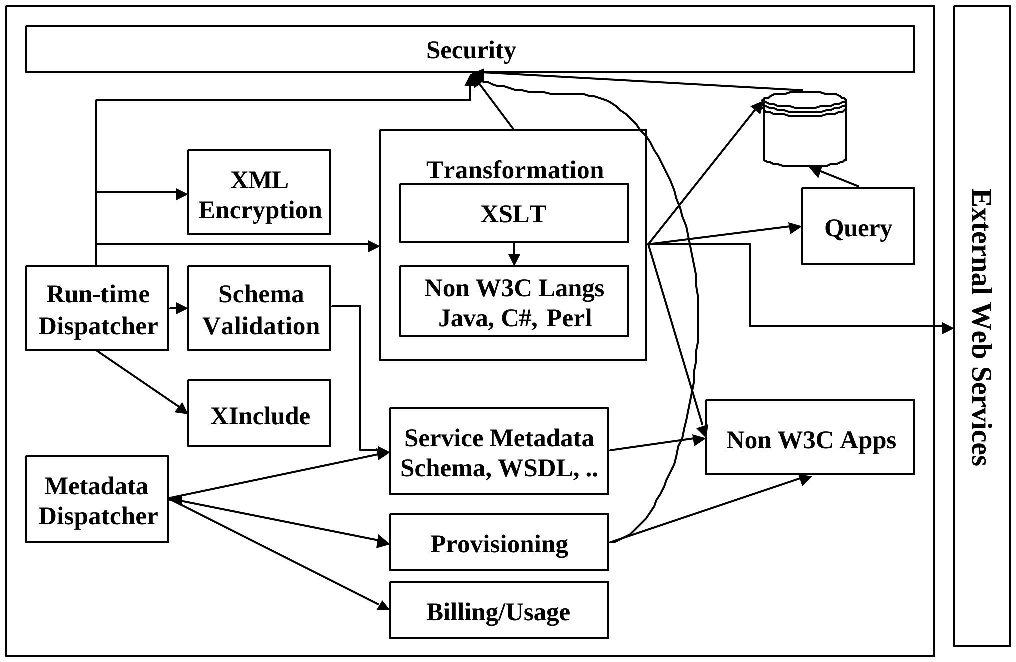
We deliberated on proposing a position that the W3C should mandate or allow XPSDL binding registries (ala here is the official XPSDL Java 1.3 binding) but decided to defer this to a chartered working group.

Web Service architecture with non-W3C specifications

The Jamcracker platform provides a number of components in the Web service architecture that the W3C is very unlikely to standardize, nor should it. Security, often overlooked in the first or second version of new specifications, is often a key factor in the success of a new platform. While the W3C has done little to standardize service level security, other groups like OASIS SAML and XACML have started to fill this need.

The Jamcracker metadata dispatcher dispatches requests to Provisioning, Billing and Usage systems. These are key components required to effectively manage a large ecosystem of business web services. The Provisioning component embodies the process of adding user, organization information, as well as Authorization for users and companies to use services. Our approach is called ITML Provisioning. The Billing and Usage component provides usage information about services consumed by users and organizations, rating of the services, and the charges for the users and organizations.

As rhetoric about dynamic discovery of services rises – what we call the "7-11" model of services - it is our belief that the security, provisioning and billing components comprise such tremendous diversity that seamless discovery and invocation will be many years away. The components of trust, user information, and billing are essentially complex long-standing business relationships, not technology relationships. The ASP and ASP Integrators – those of us who earn actual revenue on web services – have learned that these details and relationships are crucial for success, and cannot be glossed over or avoided. Ecosystems and markets will emerge that share common views of these relationships. We explicitly wish to avoid technology standardization in the area service discovery because each ecosystem will evolve differently. It is likely that standardization – *at this time* - of registries and repositories would preclude innovation in the ecosystems.



Position #5: Standardization of security credentials and results of authentication and authorization could be chartered as a W3C working group. We are impartial on this matter as we are actively involved in SAML.

Position #6: We do not believe that the W3C should charter a metadata registry and repository working group at this time. We are convinced that the attempted standardization of registries and repositories interfaces is fraught with peril, has never been successfully accomplished despite repeated attempts, is unclear about its viability, and is unproven in implementation. To be clear, we believe that communities will discover businesses and business processes and we fully support and participate in the UDDI efforts, insofar as we are permitted. Given the number of clear directions we wish to suggest to the W3C and our limited space, we choose not to explain this position further.

Conclusion

The Jamcracker position paper offers a number of clear positions for the W3C. They range from completing the XML infrastructure to standardizing interface definitions including non-Web Service interfaces and finishing with suggesting no work be done on registry/repository standardization at this time.

References

ITML Provisioning – http://www.itml.org
SAML - http://www.oasis-open.org/committees/security/index.shtml
UDDI - http://www.uddi.org
WSDL - http://www.w3.org/TR/wsdl
Xinclude - http://www.w3.org/TR/xinclude/
XML Encryption - http://www.w3.org/Encryption/2001/
XML Protocol (XMLP) - http://www.w3.org/2000/xp/
Xquery - http://www.w3.org/TR/xquery/