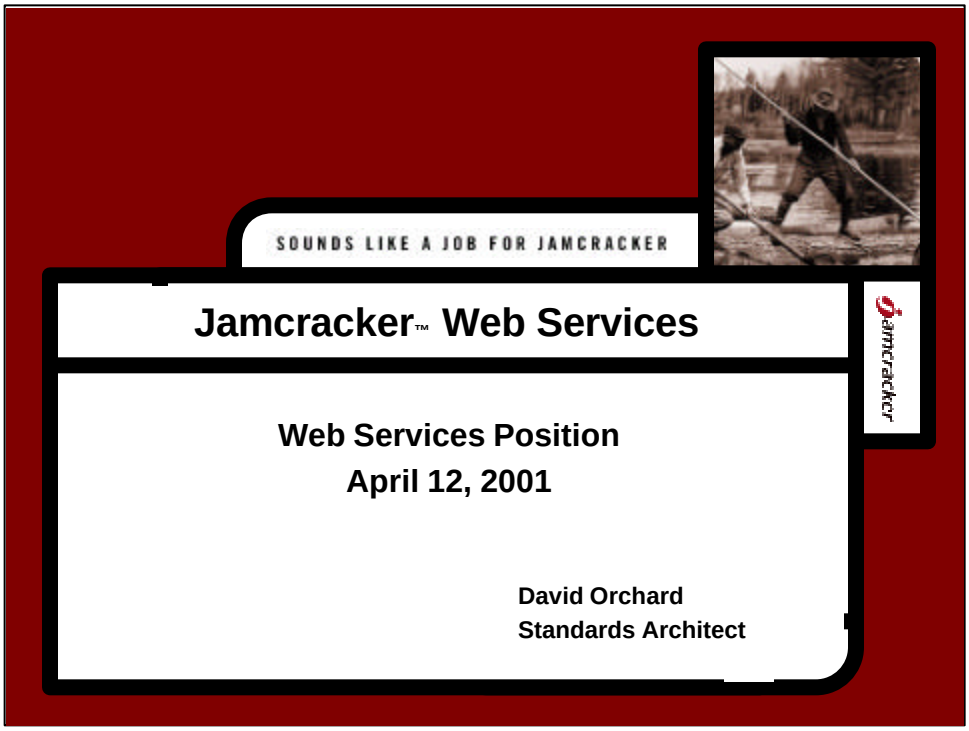

SOUNDS LIKE A JOB FOR JAMCRACKER
Jamcracker™ Web Services
Web Services Position
April 12, 2001
David Orchard
Standards Architect
jamcracker

Web Services Vision

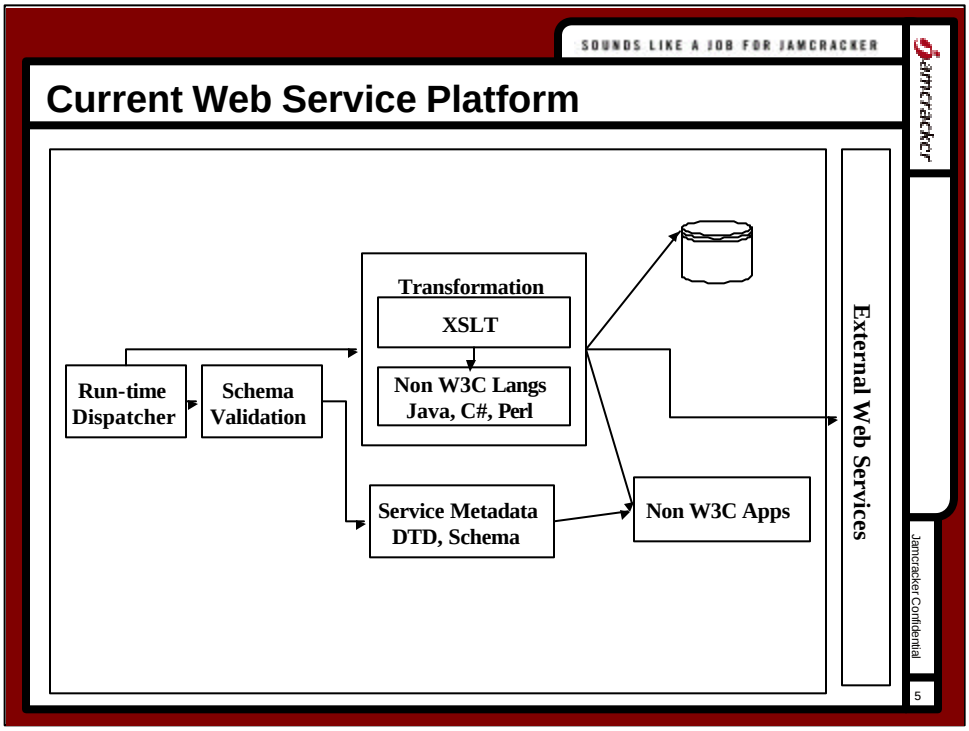
- **Provide an ecosystem of web services**
- **Integrate XML interfaces/Web Services together and with customer systems**
  - Movement from ASP Integration (HTML pages) to Web Service Integration (XML documents)
- **Integration at many levels**
  - Front-end
    - Single sign-on, Business Processes, APIs
  - Back-end
    - Provisioning, Billing, Directory, API
  - Support/Help-desk
- **Integration of Web and non-web apps**

Jamcracker vision for W3C

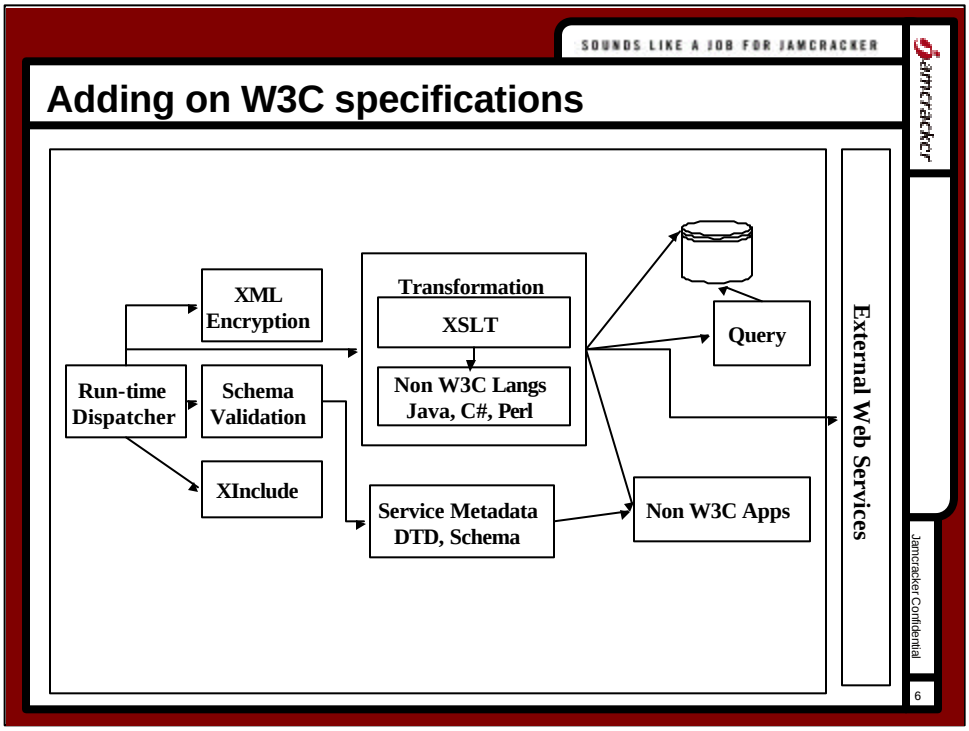
- **Create an architectural framework/roadmap**
- **Then selected pieces in order**
- **Always solve pain rather than guess pain**
- **Clean up the integration afterwards**
- **Ensure strict conformance**
- **Allow for non web and non XML world**
  - >75% of Jamcracker integrations are thus
- **Creating complete detailed architectures doesn't seem to work**
  - "Top down"
- **More of a middle-out approach**

Platform evolution

- How will this evolve from a W3C and a platform provider perspective?
- Adding on current W3C specifications
- Adding likely W3C specifications
- Adding likely non W3C functionality



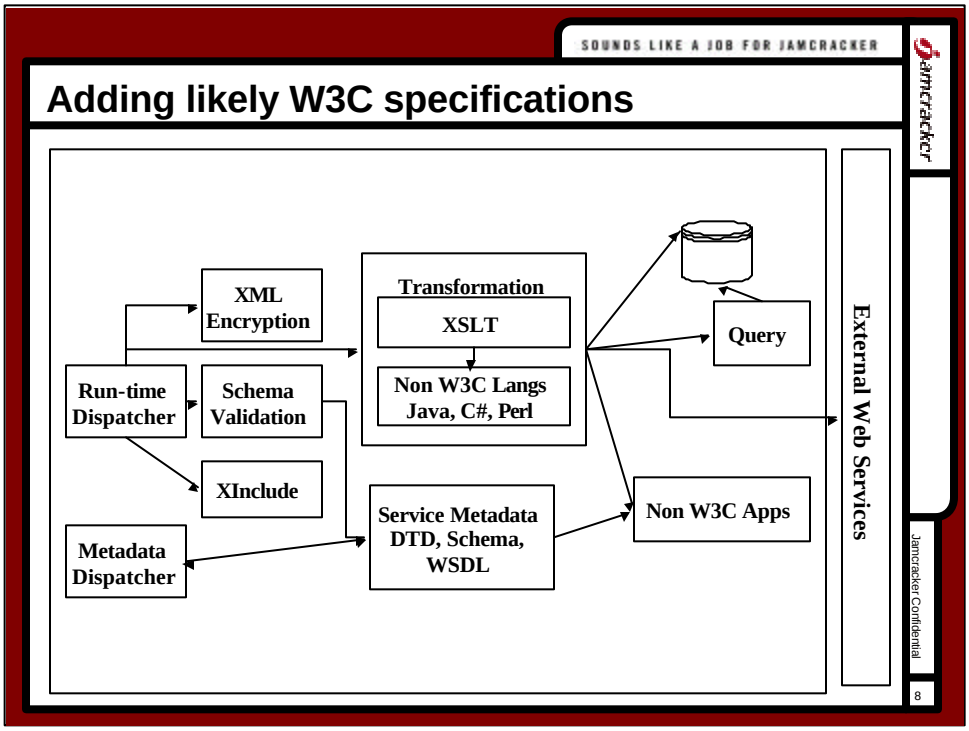
- The current web service platform provides a complex workflow of tasks based upon the receipt of a web service request.
- The input is an XML request, encoded in XML Protocol
- The runtime dispatcher, typically the first component to receive the request, will typically follow the following steps
  - Validate request type information
  - Dispatch to a transformation service
- The transformation service can be a variety of implementation 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, an Enterprise Application Integration (EAI) product.



- XML Encryption, Xinclude, and Xquery have been added to the picture
- The runtime dispatcher will typically follow the following steps
  - Decrypt request
  - Perform Xinclusions
  - Validate request type information
  - Dispatch to a transformation service
- XQuery can be be used by the transformation engine to query against documents, databases, etc.
- There is a clear and obvious need for defining the various states and processes that the xml message can be in. This has usually been called the XML Processing Model.
- Action item #1: clearly define the XML Processing model. This is a co-requisite of other web services work

Position #1: Complete XML processing model

- **A co-requisite for web services**
  - We're not finished with XML yet
    - And the entire world isn't about SOAP – yet.
  - Added many specifications that need a model
- **Required: Defined default order**
  - XMLP->XMLE->DTD->Xinclude->Schema->XSLT->Xquery->Xlink ?
- **What does http://x.org/user.xml#a refer to?**
- **In future**
  - Configurable order
    - Ie, include copyright document; post XSLT
- **Optional: Overall Architecture**



•It is likely and suggested that the W3C commence work on standardizing service metadata. A first form of this is WSDL.

•In this diagram, the metadata dispatcher is the software component that dispatches requests outside the service context. An early form of this is UDDI with it's form of requests. Typical operations are searching, adding, modifying metadata entries.

•It is clear that the metadata dispatcher will probably follow the same process as the run-time. The metadata dispatcher will be a run-time for metadata.

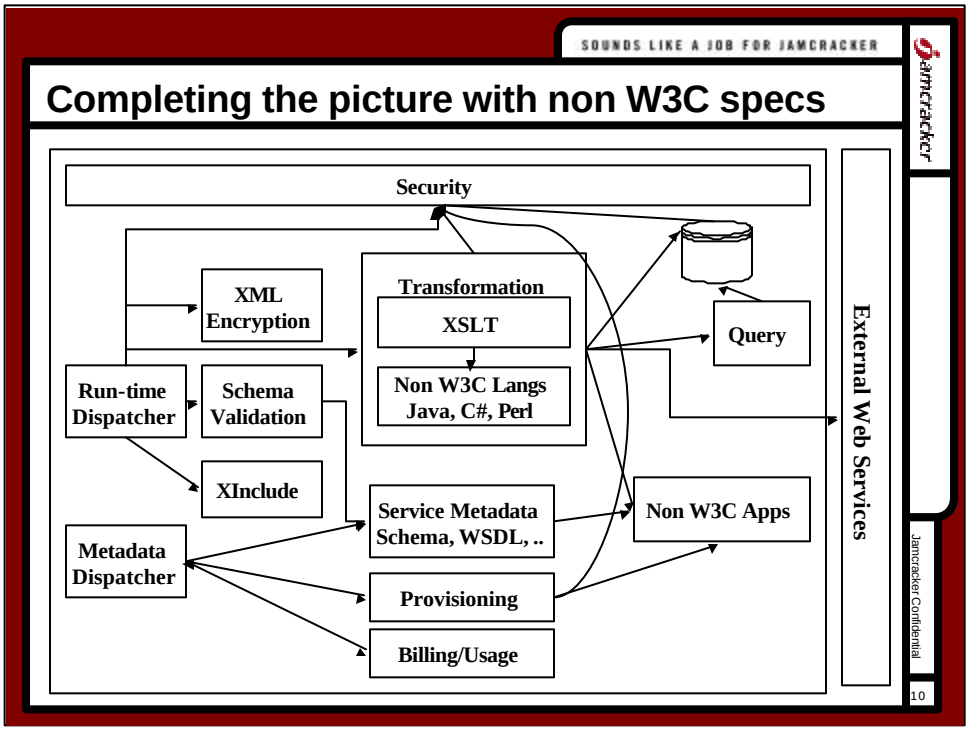
•The metadata dispatcher performs queries against all the services that are registered to it. The services can be web services, defined through WSDL, as well as non web services.

•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.

•Action item: non W3C service metadata be representable in any W3C metadata service definitions

Positions #2, #3, #4

- **Do Service Definition Working Group**
- **Loosely couple to WSDL**
- **Must be very friendly to non-W3C Specs**
  - W3C specs = XMLP/SOAP; XSLT; HTML GET/POST
  - Most of our integrations are with third-party languages
    - Portal - html screen scraping; xml
    - EAI – C++; Java; COM; cobol;SAP; …
    - B2B – RosettaNet;XML
- **Additional info:**
  - WSDL simply a step towards choreography
  - Choreography WG?



•The emerging web service platform provides a complex workflow of tasks based upon the type of request, either a web services request or a web service metadata request.

•There are many areas 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 determiner of the success or failure of a new platform. While the W3C has done little to standardize service level security, other groups like OASIS SAML and XACML have started up to fill this need

•The transformation service often performs additional Security policy verifications, such as EJB or COM+ security

•The Metadata dispatcher also dispatches requests to Provisioning, Billing and Usage systems.

•The Provisioning component embodies the process of adding user, organization information, as well as Authorization for users and companies to use services.

•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.

•The emerging web service platform provides a complex workflow of

Positions #5, #6

- **Security required**
  - We are working in OASIS on Security – SAML
  - Single sign-on/ sign-off ecosystem
  - Can be W3C WG or liason
- **Do not standardize Registry/Repository yet**
  - Technology unproven
    - Xquery and Choreography/WSDL probable co-reqs
  - Oft quoted UDDI model is not major pain point
    - Provisioning, Billing, Security, SLA, etc.

Provisioning: Complexities

- **Most Services require service specific provisioning information**
  - User: social security number, Locations, frequent flyer numbers, meal preferences
  - Application: Mailbox size
- **But constrained by company specific info**
  - Jamcracker has locations “Cuper”, “phx”
- **Intersection and union of constraints won’t just be “discovered”**
  - Humans must determine and program the semantics
- **Also: how does customer enter provisioning data?**
  - Either lazy or eager data entry
    - Bulkload, email, web pages.
- **Jamcracker has a Master Provisioning Schema**
- **And ecosystem of provisioning web services**
  - No duplicate data entry, automated provisioning, user id mgmt

Summary

- **Discuss Emergent Web Service Architecture**
- **Positions:**
  - Close the loop on XML specs
  - Do Web services Definition, probably including choreograhy
  - Don’t standardize UDDI just yet.
    - Too much other stuff to do
- **Web services marketplace is much more human-centric and harder than uddi evangelism would indicate**