

A Web Services Manifesto

Phillip J. Windley, Ph.D.
phil@windley.com
www.windley.com

The
Windley
Group



The Williams Family

Life Event: *Moving to Utah*

- Change of address
- Register car
- Register to vote
- Enroll child in school
- Bussing
- City services
- Health information
- Child safety
- Check the commute
- Tax information



Federating Services

Moving to Utah

- Real estate
- Taxes
- Register car
- Register to vote
- Enroll child in school
- Bussing
- City services
- Utilities
- Health information
- Banking
- Child safety
- Change of address
- Check commute

Child entering School

- Health information
- Grades
- Tuition and fees
- Books
- Child safety
- Bussing
- Federal programs
- Check commute

Shared services

Private services

Legacy Data

- Governments and other organizations control vast data resources
- That data is held hostage in disconnected, legacy data resources
- eGovernment requires that we free *data* from siloed systems and legacy platforms

Web Services

- Start Web services now:
 - ◆ Incrementally expose your data
 - ◆ Incrementally expose your APIs
- The more data and APIs that you expose the greater the potential interoperability
- Small marginal cost and high return, but...design is important

Design Principles

1. Every data element and collection is a resource
2. Every resource should have a URI
3. Cool URI's *don't* change
4. Preserve the structure of data until the *last possible moment* (i.e. return XML)
5. Make XML Schemas available *online*
6. Data queries on existing resources should be done with a GET
7. Use POST to create new resources

Design Principles (cont)

8. Document your service API using WSDL, WRDL, or some other standard
9. Advertise the presence of the data using WSIL
10. Adhere to data standards such as RSS where available
11. Use Metadata (RDF) for XML
12. Use HTTP authentication as much as possible
13. Make data available in multiple flavors (XSLT)

A Tale of Two Websites

- Two applications:
 - ◆ HHS has an online application that allows users to check data on nursing homes
 - ◆ Utah has an online professional licensing application
- Designs are consistent with many of these principles
- Do not yet expose data as XML
- How might they be used together?

Serendipitous Applications

- Small, scripted aggregations lead to serendipitous applications
- Example: Udell's Library Lookup

Web Services

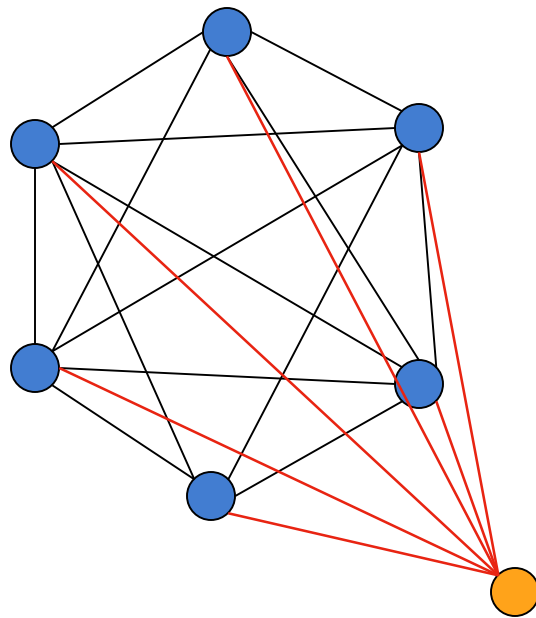
- Web services are self-contained pieces of code with three distinguishing properties:
 1. Communicate in an interoperable XML protocol, such as SOAP.
 2. Describe themselves in an interoperable XML meta-format, such as WSDL.
 3. Federate globally through XML based registry services, such as UDDI.
- Not defined in terms of SOAP, WSDL, and UDDI.

Using Web Services

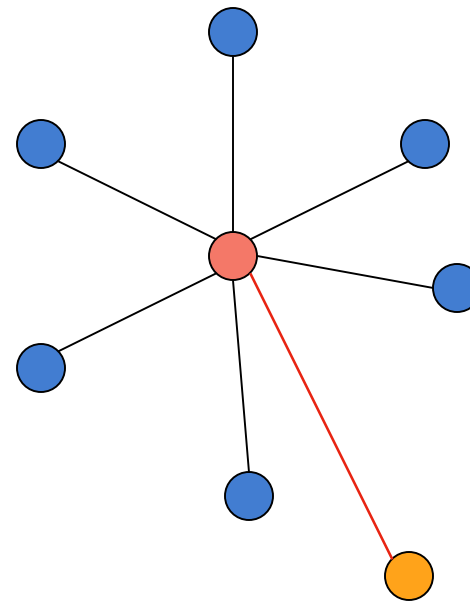
Examples

- Common Payment Gateway
 - ◆ Web services creates easy interfaces
 - ◆ No need for multiple SDKs
- Criminal Justice Network (CAD)
 - ◆ Computer Aided-Dispatching
 - ◆ Networking these systems

Need for a Broker



Without Broker:
 N^2 connections



With Broker:
 N connections

Web Services Brokers

- Connecting with outside parties is hard
 - ◆ Security models
 - ◆ Transport mechanisms
 - ◆ Semantic and syntactic mis-match
 - ◆ Trust
- WS Brokers solves these problems
- Ex: Grand Central Communications

A Word of Warning

- Good architects should do everything they can to avoid data serialization.
 - ◆ Web services is nothing *but* serialization.
- When serialization cannot be avoided, it can be mitigated through caching in some cases.
 - ◆ SOAP over HTTP makes caching difficult (uses POST).

Summary

- First steps:
 1. Don't let the hype scare you
 2. Don't try to figure it all out first
 3. Adopt simple principals
 4. Jump in and do something
- The keys are
 - ◆ XML
 - ◆ Incrementally exposing data and APIs

For More Information

- phil@windley.com
- <http://www.windley.com>
- My paper on table and at above address
- <http://www.xml.gov>
- <http://xml.amazon.com>