East-End Cockney and Web Services: It's the Semantics Silly!

My aunt Edna, who is very old, but very independent, experienced a tragic accident this past Easter: her garbage disposal and kitchen plumbing was clogged full of carrot rinds and other trimmings so that the dishwasher, sink, and bathroom unmentionables were flooded. Being a good nephew, and mostly because I was on the phone when she and her friends were trying to fix the problem, I offered to help out. One little problem: aunt Edna lives in London, I live in San Francisco. No worries I thought, this is the information age, and I can solve this challenge remotely.

With Edna still on the phone, I do the crooked neck dance over to my DSL enabled computer and proceed to go to Yahoo! Yellow Pages. Switching to the UK directory I search for "plumbing" and get a list of London area plumbers. After scanning through some of the advertisements, web sites, and general listings, I decide on one to call. At this point I have to let aunt Edna off the 'phone, she wasn't hysterical yet, but she was clearly near panic!

Telephony is an amazing thing. I simply dial the number of an East End plumber from San Francisco and, through all that switching, routing, and trans-Atlantic cable, somebody answers on the other end. This is as far as great technology like the Internet and telephony could take me – at this point I entered into this convoluted conversation of which I could barely make out a single word. The cockney (speech of Londoners) vocabulary, British slang, and colloquialisms of plumbing, repairs, and money were nearly impossible to decipher for a born and bred American.

Dynamic Discovery and Connectivity

Web services are like the technology of the Internet and telephony in this situation. Using the Yahoo! Directory was a bit like using UDDI to discover a new service: I had to know some things about basic taxonomy to search for and look up services, but I found a good list that met my criteria. The WSDL standard is a little like browsing through those advertisements, web pages, and store descriptions trying to find someone who had a service I needed and met some special criteria of mine. Finally, SOAP and HTTP together are like the telephony protocols that run over fiber, enable intelligent switching and routing, and are universally understood by handsets all over the globe.

This kind of dynamic search, discovery, and service lookup capability that web services provides is a great leap forward for solving the persistent problems that B2B and EAI solutions have been attempting to solve for years. Web services promises to make IT professionals' lives much easier by agreeing on a common, non-proprietary structure for communication protocols, registry APIs, and service descriptions. These capabilities will eventually enable a radically new kind of architectural approach, at some point in the future, for building and deploying software systems. In the near term, web services promises to make application integration easier for IT professionals because they can buy software products that don't lock them into a particular vendor strategy.

However, the hype surrounding the capabilities of dynamic discovery is greatly overblown. Web services is excellent at providing the infrastructure and common standards for dynamically discovering and connecting to previously unknown software systems, but ultimately, and you must've guessed it based on the title, real genuine communication is all about the intended meaning of the information. I was able to discover, evaluate, and connect to that British plumber in less than three minutes (including browsing around for a while) – but once I got on the phone with this bloke from the East End I couldn't seem to describe the need. He was asking me about the "loo" and wanted me to tell him if the "bog roll" had gone and clogged things up a bit. He eventually advised me to "naff off …"!

With web services there is an assumption that either (a) everyone using it will be speaking the exact same language and dialect (strict ebXML, RossettaNet, etc.) or that (b) some kind of

translation service will have to be coded to get things to communicate well. Well, where's the value in that? Not to take anything away from the brilliance of loosely coupled integration – being able to discover, evaluate, and exchange messages with anyone in real-time is amazing! But, if we're still forced to bind our messages together with code, or agree to speak about everything the same way, then we're stuck still with most of the same problems that plagued the EAI marketplace for so long – failed projects, misused and abused standards, and misunderstood vocabularies.

Steve Vinoski, Chief Architect at IONA, says it best, "Dynamic discovery is a big part of the unfortunate hype surrounding web services. While many claim that supporting dynamic discovery through systems such as UDDI is trivial, it's actually a difficult problem. It requires clients and services to share common definitions and understandings of the concepts (the ontology) involved in any web service through which they hope to interact. Without a shared understanding of the concepts, metadata, and semantics associated with a particular web service, a client cannot hope to dynamically interact with it in the correct manner."

Today, web services promise to get us out of the vendor lock-in scenario and free us from the old "integrating the integrators" problem. But, when we get right down to it, web services are leaning very heavily on the old standard vocabularies solution to the age-old problem of disparate semantics. These vocabulary standards will no doubt be significant for some industries. But the notion that a single set of standards will enable businesses everywhere to speak precisely the same vocabulary is as misplaced as the idea that EDI solved the integration challenge 30 years ago. At best, the lessons we've learned from industry's work with EDI have shown us that (a) defining standard vocabularies are difficult and time consuming, (b) once defined, standards do not adapt well, and (c) usually people don't implement the standard correctly anyway.

Dynamic Collaboration and Understanding

Enterprise Information Interoperability (EII) is an emerging category of methodology and tools targeted on solving the problem of disjointed vocabularies, data definitions, terminology and world-view in enterprise IT systems. EII is the missing link in web services. It allows engineers and business people to build loosely coupled information webs that can dynamically enable rich, semantically precise, collaboration and understanding – without the need for custom code or standardized vocabularies. Fundamentally, EII solves a completely different set of problems than either web services or B2B/EAI products – focusing on the information itself rather than the discovery, connectivity or system interfaces of integration.

The roots of the EII solution are also completely unrelated to EAI and web services technologies. EII technology derives from parts of the work done in the artificial intelligence (AI) community dating back to the early 1970's. At that time, computer scientists were grappling with issues of knowledge representation in digital systems, attempting to solve the problem of how to represent human knowledge digitally. Today's typical EII solution has three key characteristics that are both reminiscent of old school AI technologies and eerily similar to Tim Berners-Lee's effort with the Semantic Web proposals at the W3C.

1. **Semantic Mediation** – Using ontology, a model that makes concepts explicit, as a mediation layer in order to abstract particular data terms, vocabularies, and information into a sharable and distributable model. This is akin to some of the old notions about creating a "model driven enterprise" – using core information models as a lens to refract enterprise data in whatever form required.
2. **Semantic Mapping** – The ontology is only as good as the quality of the map that associates the enterprise data to it. Mapping is the core technique that eliminates custom code and preserves the native semantics of the data. Mapping, for EII companies, accounts for much more than simple many-to-many data formatting rules – it is how the semantics are captured, aligned, and structured in relation to the data itself – creating information out of "dumb" data repositories.

3. **Context Sensitive** – As linguists know very well, the meaning of any data is always bound to a particular perspective, or context. Thus, any EII solution set must accommodate the fact that the same data can mean many different things from different viewpoints. Typically, the business rules, context definitions, and environmental metadata are captured and stored during the mapping process, making them reusable in any runtime server process.

The actual enterprise software tools that are used in middleware environments vary by vendor, but typically include at least a server-based process that can mediate real-time and batch-oriented messages and data exchanges. In addition, the design time GUI tools will usually include a mapping tool that can build the semantic and contextual structures that link together ERP, PDM, EDI, XML and other data formats to the ontology. By focusing on the elements of interoperability that pertain to vocabularies, taxonomy, and data meanings, the EII solution crosses established boundaries of web services, B2Bi and EAI solutions – working equally as well for internal and external information exchanges.

However, don't look to EII as a silver bullet solution to every EAI, XML, and other integration shortcoming. EII technologies are a component of an overall solution. The business needs of interoperability still dictate that software adaptors, message transport services, and business process management techniques must be accounted for. This fact is exactly why the EII space dovetails so well with web services technology: while web services create a loosely coupled framework for business process, connectivity, and discovery, EII creates a loosely coupled framework for the core information exchange that is required during the overall process. In addition, there are significant advantages to using an EII solution as a compliment to more traditional integration technologies such as EAI, ETL, and composite technologies (CORBA, EJB, and COM).

EII is not a silver bullet solution for long implementation times during complex integration projects. This is a problem of simple math: the typical integration project spends the vast majority of time and resources on analysis, understanding business rules, understanding the data, and engaging the appropriate subject matter experts. No technology solution can speed up this fundamental human activity. However, EII does offer a distinct advantage here. Once the analysis has been done, rather than embedding all that knowledge in custom code or proprietary scripts, the EII solution will allow the project to capture that knowledge in the semantic and context maps, thereby making the work reusable for all those who follow. The essential human expertise, rules, and data understandings will have been formalized in a way that is long-lived. Each additional system that joins an EII community is incrementally cheaper to bring online – versus exponentially higher costs for adding or removing systems in more traditional integration solutions.

Technology visionaries all over the world have reached the same conclusions: ontology based semantic mediation of disparate information resources is the most viable alternative to more of the same old thing – custom code and monolithic standards. It will be an essential part of all interoperability solutions in the very near future. A union between the capabilities to dynamically discover services and connections (web services) and the ability to dynamically discover shared concepts and meanings (information interoperability) will truly be a match made in heaven!

A Match Made in Heaven

Web services stands to benefit from this technology greatly. Imagine if a web services exchange could not only provide UDDI discovery of location, but also of vocabulary, concepts, and of context maps that provide rules for how to move between vocabularies. Imagine if WSDL provided descriptions of those ontology and maps – so that any requesting systems could check to see if they shared a point of reference or could dynamically discover maps that enabled rich communication to begin. Finally, as with all web services, XML, SOAP, and HTTP could be the standard envelope and protocol of exchange, but the message contents could be EDI, ASC X12,

EDIFACT, flat files, delimited files, XML documents, proprietary messages, or whatever – and you would still be able to discover, evaluate, and begin to use these services dynamically.

This is the promise of web services, this is the value – but web services alone don't get you there without an EII partner. A complete solution that is both loosely coupled and inexpensive to deploy and maintain, must have at least three core elements:

1. A sophisticated and loosely coupled physical infrastructure – Web Services
2. A sophisticated and loosely coupled information infrastructure – EII
3. Quality analysis and planning – the human element

Lest we forget that all serious and difficult problems, including interoperability and integration, are human problems. Ultimately, the disjointed vocabularies of disparate computer systems are the results of differing culture, ideas, business processes, folklore, and viewpoints. An EII solution focuses on the needs of the users and organizations that are at the core of any sophisticated information exchange.

One day we may have a universal language interpreter for phone systems, but until then whenever aunt Edna needs long distance support for her local problems, I'm going call up my cousin John in Essex and ask him to take care of it! In the meantime, we should take advantage of enterprise information interoperability platforms to take care of these semantic, vocabulary, and contextual problems of system-to-system communication with web services – it's the semantics silly!