

Service-based IEMML

Whitepaper

Andrew Roczniak
March 2008
Version 1.0

1- Introduction	3
2 – Web Services	3
3 – Expected Benefits.....	4
4 – Technical Details.....	5
4.1 - Comparison with other technologies.....	6
4.2 - Standards status.....	6
5 – Requirements	6
5.1 - SOAP vs. REST	6
5.2 Inter-Service Exchanges	7
5.3 Service Description	7

1- Introduction

Leveraging the assets of the IEML project, namely IEML syntax, IEML parser and ieml.org website, and building on the open-source nature of the project, this whitepaper outlines an approach to create an ecosystem of researchers, users and developers with the goal of facilitating the collaboration within the existing IEML users' community and encouraging its growth.

The inspiration for this approach comes from the Web 2.0 domain in general, and opening up of social networks in particular. Acknowledging the power of individual creativity, social networking leaders such as FaceBook and Google recently allowed users to create widgets, tools and applications that operate on the underlying social connections. Similarly, with the IEML syntax and parser at its core, the following discussion outlines the basic infrastructure that will allow new and existing users to design and share services that operate on information encoded with IEML.

The two main objectives of this approach are to reduce the effort necessary to create IEML-processing software for current users as well as for users considering IEML for their own specific domains, and secondly, hiding the computational complexity such that users operate only at the IEML conceptual level.

2 – Web Services

In a very short time span (Internet time), the Web has become the universal platform for the distribution of all kinds of multimedia resources published and consumed primarily by humans. The Web Services (WS) technology reuses and extends the Web's successful technologies to standardize machine-to-machine interactions on a global scale. In essence, the technology consists of a set of protocols and languages like SOAP, WSDL, XML used to describe and convey service requests and results, or REST, an architectural style.

The WS technology is becoming the prevailing middleware standard for the implementation of Service Oriented Architectures (SOA) built as a composition of services made available over a network. IEML project adopts these SOA principles, and Web Services as a general exposure mechanism independent of each potential user's executing environment.

3 – Expected Benefits

End users and organizations have the direct benefit that systems built with web services foster the rapid creation of new services in an open innovation environment. So, for the end user, more services are available at a faster pace and at a lower cost. This is what we are seeing today with the proliferation of “mashups”.

WS technology provides an open, standard-based, platform-agnostic middleware environment that yields three key benefits. First, Web Services foster reuse opportunities. Indeed:

- Reusing services having standard-based interfaces is made easier thanks to the native support of these standards in leading development environments.
- Standards are platform-agnostic, which fosters reuse in heterogeneous software and hardware environments.
- WS standards provide reusable solutions to a set of common problems across application domains.
- WS standards improve the interoperability between services, and thereby lower barriers to reuse.
- Implementations relying on WS-BPEL (Web Services Business Process Execution Language) may be deployed on different software and hardware platforms without modification.

Second, Web Services lower integration costs. Indeed, the standardization of interfaces results in:

- Reduction of efforts spent on integration,
- Increased availability of a skilled workforce, focusing on a limited set of standard technologies,
- Commoditization of middleware and tools,
- Reduction of vendor lock-in risks,
- Availability of open-source solutions.

Third, Web Services increase business agility. Indeed:

- As integration efforts are reduced, and more services can be reused, development cycles can be shortened.
- Web services leverage the Web technologies, and can become the ubiquitous, world-scale standard for machine-to-machine communications. IEMML community will be able to orchestrate services spread all over the world and rely on heterogeneous platforms.

- Service interfaces can be standardized and opened to an ecosystem of partners. Third party services may then be selected and changed more easily as long as they provide similar service interfaces.
- Business Processes can be automated using high-level, standard, Business Process Modeling languages (e.g. WS-BPEL) and open-source tools. This results in shorter development cycles and improved productivity.

Last but not least, Web Services are universally adopted for the transformation of Information Systems towards Service Oriented Architectures, which further contributes to delivering these three benefits.

4 – Technical Details

How does it work? Web Service technology is made of:

- A core messaging layer: web protocols and languages standardizing the transport (e.g. HTTP/S), the structure (SOAP) and the syntax (XML) of information exchanges within the context of a service delivery.
- A modular set of (sometimes overlapping) standards adding all kinds of generic capabilities that may be required to deliver services such as security, ontology management, reliable messaging, support for transactions, publish/subscribe mechanisms, semantic process modeling.
- A set of standards for the description, the registration and the discovery of services.

Two major standards are defining Web Services: SOAP and WSDL. SOAP messages carry service requests and responses between a service requestor and a service provider. They are XML documents structured in two parts: the optional header that contains meta-data used by SOAP intermediaries or ultimate receivers to process the message and the body that contains the XML encoding of the actual service request or response.

WSDL provides a standard XML description of Web Services. It is a major component of service contracts exchanged between a service provider and service consumers. With this description, leading Software Development Environments can automate the production of code required to invoke or deliver the service.

4.1 - Comparison with other technologies

Web Services technology is often compared with CORBA, as it actually re-implements most of CORBA's features (transport of service requests/responses, referencing of resources, interface description language, standard services, ...) using web-like protocols. CORBA (Common Object Request Broker Architecture) has been the de-facto standard middleware used for client/server interactions in the past decade. Web Services provide basically the same kind of features, but rely on web-originated languages and protocols (e.g. HTTP, XML).

4.2 - Standards status

Web Service technologies are standardized within two main bodies:

- the World Wide Web consortium (W3C)
- the Organization for the Advancement of Structured Information Standards (OASIS)

The Web Services Interoperability (WS-I) organization is another important organization chartered to promote Web services interoperability. It has developed a Web Service Basic Profile that addresses more than 200 interoperability problems already.

5 – Requirements

5.1 - SOAP vs. REST

Should we use SOAP or REST?

Firstly, there is already lots of debate on this question; we are definitely not the first:

<http://tomayko.com/weblog/2008/01/13/lying-through-their-teeth>

A quick overview of what it is and who is using what, see for example:

<http://www.petefreitag.com/item/431.cfm>

My arguments will be based in part on:

<http://blogs.iona.com/vinoski/archives/000152.html>

REST-related links:

<http://www.oreillynet.com/pub/wlg/3005>

<http://www.flickr.com/services/api/>

<http://www.manageability.org/blog/stuff/soap-is-dead>
<http://blogpro.toutantic.net/2007/11/23/ws-bashing/>

The purpose of a service-oriented architecture for IEML, from my point of view, is to make interaction within IEML community simpler, and to make joining our community easier for interested collaborators.

The ieml.org website will for now be the directory of available services, as well as a repository of those.

What should the service interface be? Designer of the service will have their own preferences, as well as the consumers of that service. The ieml.org website is neither the developer, nor the consumer...

So, just like Amazon or eBay or others, it is better not to presume what is best for the clients, but allow them to choose what they feel comfortable with. In short, then, we should allow either REST or SOAP.

5.2 Inter-Service Exchanges

IEML-encoded information exchanged between a service producer and consumer must be a valid parser output. Services that receive the whole IEML-encoded information may verify the validity of that information, but services that operate on streams of IEML-encoded information may not do so.

5.3 Service Description

The description of a service should include a human-readable description in English (or French), and must include an IEML-encoded description (human and machine-readable).