Towards a Common Platform to Support Business Processes, Services and Semantics

Baba Piprani

MetaGlobal Systems, Canada
babap@attglobal.net

Abstract. The search for the Holy Grail in achieving interoperability of business processes, services and semantics continues with every new type or search for the Silver Bullet. Most approaches towards interoperability either are focusing narrowly on the simplistic notion using technology supporting a cowboy-style development without much regard to metadata or semantics. At the same time, the distortions on semantics created by many of current modeling paradigms and approaches – including the disharmony created by multiplicity of parallel approaches to standardization – are not helping us resolve the real issues facing knowledge and semantics management. This paper will address some of the issues facing us, like: What have we achieved? Where did we go wrong? What are we doing right? – providing an ipso-facto encapsulated candid snapshot on an approach to harmonizing our approach to interoperability, and propose a common platform to support Business Processes, Services and Semantics.

Keywords: Services, SOA, Business process, Semantics, Conceptual Schema, ORM, TR9007.

1 Search for Interoperability

Interoperability---meaning, diverse systems and organizations working together---is the much sought after goal, IT shops are always striving and searching for IT Nirvana, in the guise of re-usability, common functional components, Services Oriented Architecture, the now defunct term Grid Computing (or is now re-named Cloud Computing…) , and so on. The term "interoperability" is often used in a technical sense i.e. systems engineering, or, the term is used in a broad sense, taking into account social, political, and organizational factors that impact system to system performance. Syntactic Interoperability is when two or more systems are capable of communicating and exchanging data using some specified data formats or communication protocols, e.g. XML or SQL standards---much like interoperability of rail, which have common parameters like gauge, couplings, brakes, signaling, communications, loading gauge, operating rules etc. Syntactic interoperability is a .pre-requisite to achieving further interoperability.

Semantic interoperability is the ability to automatically interpret the information exchanged meaningfully and accurately in order to produce useful results as defined by the end users of participating systems, all the participating sides must defer to a

R. Meersman, P. Herrero, and T. Dillon (Eds.): OTM 2009 Workshops, LNCS 5872, pp. 629–638, 2009.

common information exchange reference model. The content of the information exchange requests are unambiguously defined: what is sent is the same as what is understood.

Quoting the 'Helsinki Principle' from ISO TR9007 [1]

"Any meaningful exchange of utterances depends upon the prior existence of an agreed set of semantic and syntactic rules. The recipients of the utterances must use only these rules to interpret the received utterances, if it is to mean the same as that which was meant by the utterer."

2 Oh What a Tangled Web We Weave….Processes, Services and SOA

Introduced in the early years of file based systems to model behavior and to model dynamics of the enterprise, early incarnations of Process Modelling progressed from modeling decompositions of business processes, data flows for the IT crowd, and including work flow. With more than a dozen notations, process models were largely a mixed bag serving multiple clientele---business process flow, IT processes analysis, and IT implementers looking for incorporating re-usability. What was missing, and still is, is the formal bridge into data semantics for the automatable processes. More emphasis is being put on the sequencing and orchestration, making the processes an ad hoc grab bag of Picasso style compositions of complex and elementary processes woven together.

Enter "services" as in a services oriented architecture (SOA) with a web based focus to provide a new "processing" focus. The infrastructure crowd already had their infrastructure services, and so did the application folks with their application services. Yet the same term "service" essentially is being used to "service" multiple definitions. The new business model that had difficulty gaining acceptance with the term grid computing now became Cloud Computing, with new paradigms SaaS (Software as a Service), PaaS (Platform as a Service)…being part of XaaS (Everything as a Service)---based on the concept of providing encapsulated functionality via re-usable software components.

With the definition of "service" covering a broad base ranging from Information science e-services, infrastructure services, application or process based services, web based services, and so on---the report card for a success rate for a service is not looking good.

Firstly, the real story on re-usability of software components. Though Objected Oriented programming was touted as being the ideal platform for hosting re-usable objects, "real" re-usability was not achieved with much success there [2].

"Services" was touted as offering re-usability. Here again, about 1 in 5 services achieved the status of being 're-usable' per se. Was it worth investing heavily in SOA to achieve a 20% re-usability factor? [2] [3] On the flip side, there exist several cases where re-usability has been achieved on some basic categories of services, like common services e.g. looking up the qualifications of organization personnel, notification of email etc.---invoked by cross-functional business units, or data transform or cross reference services etc. Suffice it to say, re-use is not a "build it and they will come" solution (quote from movie "Field of Dreams").

A major factor in the 'Service" equation was the lack of a consistent definition semantics, and, that it is essentially addressing the "rail track" interoperability---a syntactic connect via a defined interface, defined protocol, message format etc, but without much emphasis on application semantics. In other words while the IT departments spend a lot of money on software that hums, whirs with flashing lights, more often than not the business user community is not finding its objectives being met with a business emphasis.

Another point is that of terminology confusion. Several "consulting firms" are inter-mixing "processes" and "services". Introduce into the foray the traditional terms like "business function", "function", "business activity", "activity" and "tasks". So now we have a mélange of terms meaning that some transformation is occurring, sometimes with some business focus, sometimes with a technical interface focus, sometimes with a protocol focus, sometimes with a web focus, with no standardized engineering-style specifications.

But what is important to note is that most every definition of a service is focusing only on the interface, negotiation, and basic capability. Related work in the area of service description and registration addresses these key points in various approaches.

Take WSDL, the Web Service Description Language, an XML based language for describing Web services and how to access them. WSDL is a W3C (World Wide Web Consortium) Recommendation. WSDL essentially has a set of definitions that describe a web service using the main elements:

- <types> The data types used by the service [uses XML syntax for max neutrality]
- <message> The messages used by the service [parts, like parameters for a function call]
- <portType> The operations performed by the service [like a function library or module]
- <binding> The communication protocols used by the service [format and protocol details]
- Other elements like extension elements and a service element that makes it possible to group together the definitions of several web services in one single WSDL document.

Along comes UDDI, the Universal Description, Discovery, and Integration directory services where businesses can register and search for Web services. UDDI is a platform independent framework for describing services and facilitates the integration of business services through discovery via the Internet. Basically it is a directory of web service interfaces described by WSDL where for example airlines could register their services into a UDDI directory and travel agencies could then search the UDDI directory to determine the airline's reservation interface, that is, if the travel industry published a UDDI standard for flight rate checking and reservation.

And then there is OWL-S (Web Ontology Language for Services), as an ontology of services which users and software can discover, invoke, compose, and monitor Web resources offering particular services and having particular properties. Three main parts are described---the service "profile" for advertising and discovering services, the "process model", which provides a description of a service's operation, and the "grounding", which provides details on how to interoperate with the service

via messages. It is interesting to note that a "process" as referred to in OWL-S (atomic process, composite process) is essentially a specification of the ways a client may interact with a service, and not referring to a program to be executed---as the term is recognized in Process Modelling or IT analysis!

It is important to note that the above mentioned paradigms approach the interaction of Services essentially at the "syntactic" level.

In an effort to approach services from a semantic viewpoint, WSMO (Web Service Modelling Ontology), also a W3C submission, introduces ontologies along with goals, to interact with Web Services, highlighting the semantic description of Web Services in terms of Capability and Interfaces along with connectors between components with mediation facilities for handling heterogeneities. Ontologies are used as a data model for all resource descriptions and all data interchanged during service usage and as a central enabling technology for the Semantic Web. Web Service Modeling Ontology (WSMO) provides a conceptual framework and a formal language for semantically describing all relevant aspects of Web services in order to facilitate the automation of discovering, combining and invoking electronic services over the Web.[4]

The W3C 2005 submission of WSMO also defines a Web Service Modelling Language (WSML) for the semantics and computationally tractable subsets of the formal syntax introduced in WSMO. WSML which provides a formal syntax and semantics for the Web Service Modeling Ontology WSMO. WSML is based on different logical formalisms, namely, Description Logics, First-Order Logic and Logic Programming, which are useful for the modeling of Semantic Web services.

While the Web Service Modeling Ontology WSMO proposes a conceptual model for the description of Ontologies, Semantic Web services, Goals, and Mediators, providing the conceptual grounding for Ontology and Web service descriptions, WSML takes the conceptual model of WSMO as a starting point for the specification of a family of Web service description and Ontology specification languages. The Web Service Modeling Language (WSML) aims at providing means to formally describe all the elements defined in WSMO. [5]

3 Semantics in Services…Wherefore Art Thou?

So, why is SOA having a high failure rate—like 50% [6] [7], and with empty promises of re-usability? [2] [3] [8].

Or should the question be raised, "Semantics, for what purpose are you there?" (in contrast to the section heading above which really asks 'for what purpose' semantics (Juliette asks Romeo… O Romeo, Romeo, wherefore art thou Romeo?, Juliet really means "for what purpose?"). The services in SOA essentially are addressing the syntax part (and the semantics of the mechanics part) of the web services for messages and interfaces, and not really addressing semantics of the business requirement---in particular the semantics of the community of concepts, like say SBVR [9], ORM [10] or NIAM [11] i.e. fact based modeling. But, there is a glimmer of hope with WSMO injecting the much needed business semantics via the use of ontologies, even though the engineering style declaration of semantics, the much needed grammar and transforms appear to be inadequately defined or incomplete

within the ontology declarations---in contrast with the declarations available in fact based modeling methods.

While WSMO purports to address semantics via the ontology connection, it is interesting to note that the WSMO model for capability specification, for example, defines pre-conditions for a service as what a web service expects in order to provide its service, i.e. define conditions over its input. These pre-conditions are essentially, valid business rules that are required to be met for the service to be performed. The question is, what is the guarantee that all design architects can come up with the same required set of business rules? In other words is there a conceptual schema that these rules can be derived from? In the absence of such a formal connection, the inclusion of the rule-set in pre-conditions may not be consistent.

4 --So Long as I Get Somewhere

"Would you tell me, please, which way I ought to go from here?"
"That depends a good deal on where you want to get to," said the Cat.
"I don't much care where--" said Alice.
"Then it doesn't matter which way you go," said the Cat.
"--so long as I get SOMEWHERE," Alice…

Alice in Wonderland, by Lewis Carrol

Not being able to "engineer" services through some sort of a standard specification approach in addressing semantics---not the semantics of the service protocols or message, here I mean the semantics of the "business" --- in itself presents high risk in implementing SOA.

What then seems to be occurring is that there is a large-scale dependency and trust on the mechanics of services in SOA, meaning: provide the syntax, protocols, messages, and a service is successfully accomplished---along with flashing lights, etc. There is generally a hype-follower set that embraces any new offering that promises service nirvana.

An SOA implementation that is merely focused on IT and technology, and that does not cater to the real business need, is essentially a high risk candidate for failure. A carefully thought out metadata semantic connection in SOA---both the semantics of the service and semantics of the business---is one way towards assuring a successful implementation.

Take for example, a pre-condition (a set of semantic statements that are required to be true before an operation can be successfully invoked) definition for a Web service for booking tickets from [12] as in Fig. 1.

We see here a code snippet for a capability pre-condition for booking tickets or complete trips. Agreed, this defines a specification for the preconditions…but the question is whether the analysis process used is repeatable and, that other architects would also arrive at the same solution? In other words, the code does appear as needing to be hand-crafted for each service using the same set of facts. And, more importantly, if other services have this sort of pre-condition code built in, changes affecting any of the common rule subsets could cause a domino effect in other service pre-conditions. Then the re-usability and ease-of-specification problem has suddenly taken the form of spaghetti code, albeit in a different form---resulting in dramatic failures or inconsistencies across the board.

capability VTAcapability
sharedVariables
{?item, ?passenger, ?creditCard, ?initialBalance, ?reservationPrice}
precondition
definedBy
exists ?reservationRequest
(?reservationRequest[
reservationItem **hasValue** ?item,
passenger **hasValue** ?passenger,
payment **hasValue** ?creditcard]
memberOf tr#reservationRequest **and**
(?item **memberOf** tr#trip or ?item **memberOf** tr#ticket) **and**
?passenger **memberOf** pr#person **and**
?creditCard **memberOf** po#creditCard **and**
(?creditCard[type **hasValue** po#visa] **or**
?creditCard[type **hasValue** po#mastercard]))

Fig. 1. Service Pre-Condition example

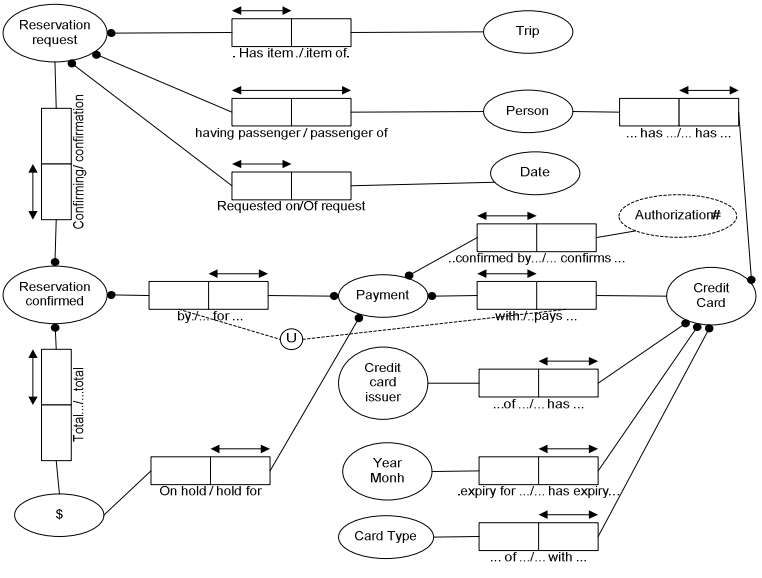

Fig. 2. ORM Conceptual Schema of Trip Reservation

Contrast this approach with metadata defined semantics in an ORM conceptual schema in Fig. 2 for a trip reservation. The semantics as defined in the pre-condition example are defined explicitly in the ORM conceptual schema, centrally, in one place only with no redundant declarations. The schema requires a reservation request must have a reservation item (trip) and one or more passengers on a given date.. A

confirmed reservation then must fulfill an existing reservation request and must be accompanied by one or more credit card reservation payments each of which must have an amount put on hold with the corresponding credit card. There is a total amount for a confirmed reservation and the sum of the amounts held on each credit card must be greater than or equal to total amount of reservation depending on the credit card company rounding algorithms.

This means that the pre-condition (with comparable semantics) as defined previously are generatable from the ORM schema, and not required to be hand-coded---which invariably leads to coding errors or inconsistencies. Many of the current service definitions are essentially hand-coded and fall into the same spaghetti coding trap, thus making re-usability and change management a nightmare. Using ORM schema and its generated declaratives is an example of meta-data driven software in action.

So, let's look at an example of a post-condition [12] as in Fig. 3.

postcondition
definedBy
exists ?reservation(?reservation[
reservationItem **hasValue** ?item,
price **hasValue** ?reservationPrice,
customer **hasValue** ?passenger,
payment **hasValue** ?creditcard]
memberOf tr#reservation **and**
?reservationPrice **memberOf** tr#price)

Fig. 3. Service Post-condition example

Examine the ORM Schema in Fig. 2. All the statements defined in the post-condition are met by the ORM Conceptual Schema. That means, the entire set of pre-condition and post-condition statements can be safely eliminated, since these rules have been consistently declared in the ORM schema and guaranteed to consistently come across in any implementation transforms, are universally applicable, and not associated with any particular service.

In practice, this entire ORM conceptual schema as shown and transformed to SQL was 100% achievable via ISO SQL[13] schema declarations automatically generated from the ORM schema converted to an ERWin [14] IDEF1X [15] based attribute data model.

Another viewpoint of a pre-condition example could be that a Service or a Process needs to be executed before a given service execution occurs. I have seen examples of such pre-conditions being stated….but is not that what Choreography is all about? But, examples abound with such redundant declarations with the very same requirement being stated in a service pre-condition as well as in a choreography model---which again contributes to failures due to inconsistencies and redundancy maintenance issues.

An item of concern is the set of semantic annotation functionalities being introduced for WSDL files via a WSDL defined semantic model, which consists of 4 parts: Input semantics, Output semantics, Precondition, and Effect (after an operation---both successful and unsuccessful). With the individual model semantics being at the service level, there does not appear to be a conceptual schema style definition model where all the rules are to be declared. For example an output semantic from one service that is feeding an input semantic of another service could have different model semantics, since input and output semantics are declared on an individual service.

It is inevitable that the more human coding that is introduced, that the code is increasingly prone to embracing failures. No wonder there are so many failures in SOA!

5 Proposed ISO Standard Work in Services for Interoperability

There is a Working Draft for an ISO Standard to address a metamodel for Service Registration [16] whose scope is to address semantic interoperability of services through achieving a common understanding between users and services for the integration of heterogeneous services. The proposed Service registration standard hopes to provide a mechanism for registering service semantics. In contrast, UDDI does not consider service semantics and only is concerned with web services.

Specifically, the scope of Metamodel Framework for Interoperabilty—Part7 Metamodel for service registration is to: Specify a metamodel for registering services that can enable users to discover appropriate services; Define the functional and nonfunctional description of services; and Promote semantic interoperation between various services. The proposed standard does not specify language specific details nor the composition of services. Fig. 4 depicts a UML diagram of the proposed ISO metamodel for service registration.

The ISO model appears to be modeling the WSMO and WSDL alignment specifications, and inherits the same deficiencies as noted.

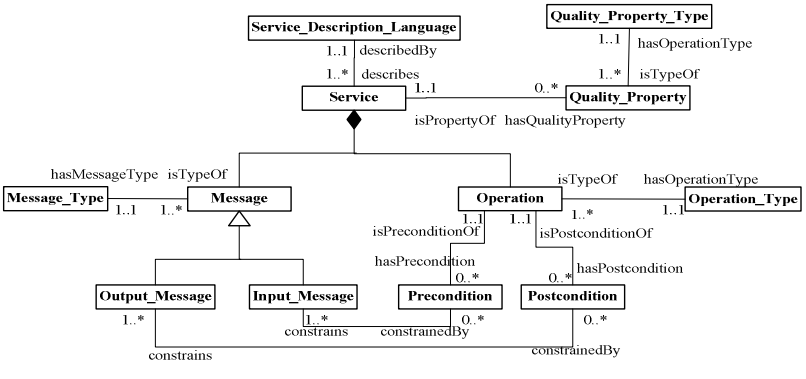

Fig. 4. Proposed ISO WD Metamodel of Service Registration

6 Towards a Terminology Based Platform Bridging Processes, Services and Semantics

What we have seen is a stampede towards finding the "silver bullet" via services. As we have seen, there is inconsistency in the definition of "services". The term is used interchangeably to mean: Infrastructure services, also Web services, also Application services, also Process, etc….There is still the 'Picasso' style analysis with no engineerable model.

Missing, is the semantics connection. In modeling semantics again, there are various incarnations: Entity Relationship (ER) that models entities and relationships, but suffers from lack of clarity and separation from attribute level semantics; Unified Modelling Language (UML) which models classes, associations and attributes, and consists of various sub-models or packages, but brings in implementation artifacts early in the modeling exercise, is essentially geared towards implementation using the Object Oriented paradigm. UML model semantics are slightly more expressive than ER, albeit at the attribute based level; Fact Oriented methods (NIAM, ORM, FOM, CogNIAM…) which have gained firmer ground in addressing semantics, are more conceptual schema oriented, are based on natural language foundation with added formalisms; with OWL and RDF applicability as this effort is in its early stages.

Recall the definition of a Conceptual Schema in ISO TR9007:1987 " The description of the possible states of affairs of the universe of discourse including the classifications, rules, laws, etc. of the universe of discourse" i.e. the Conceptual Schema covers Static and Dynamic aspects of the Universe of Discourse. Current offerings of SOA, Services, Process Modelling and data modeling overlook this important coupling. ISO TR9007 deals with propositions, i.e. Natural Language sentences in messages from the information system. We seem to have omitted this important fact in our search for semantics based interoperability.

There is a need to establish a bridge between the business model (in business-speak language) and, the implemented information systems, i.e. use of terminology for specifying and determining business static rules and business dynamic behavioral rules that drive the data models, process models and service models for an information system. Much background work has already been performed in ISO and other areas, including Fact based modelling methods SBVR as a common platform in the use of terminology as applied to communities. It is within such communities that the discovery of applicable services can be made, and definitely not at the syntactic level. A strong foundation of using established semantic modeling techniques is an absolute must for the survival and flourishing of a services architecture. The knowledge triangle [17] should be used as a fundamental start point for a platform for defining an all-encompassing solution in the area of static and dynamic behavior of an enterprise. We need to position Semantic modeling methods, process modeling and Services architectures to achieve implementations that harmonizes data semantics, process semantics, orchestration semantics and platform independence.

This paper is based on the keynote speech given by the author at the Metadata Open Forum in Seoul, Korea, June 2009 [18].

References

1. International Organization for Standardization(ISO): ISO Technical Report TR9007: Information processing systems—Concepts and Terminology for Conceptual Schema and the Information Base (1987)
2. McKendrick, J.: Pouring cold water on SOA reuse' mantra, ZDNet (August 2006), http://blogs.zdnet.com/service-oriented/?p=699
3. Linthicum, D.: Core Value of a SOA is the Ability to Reuse Services? Not a Chance (October 2007), http://www.infoworld.com/archives/emailPrint.jsp?R=printThis&A=http://weblog.infoworld.com/realworldsoa/archives/2007/10/core_value_of_a.html
4. Web Service Modelling Ontology (WSMO), http://www.w3.org/Submission/2005/SUBM-WSMO-20050603/, http://www.wsmo.org/TR/d2/v1.4/20070216/
5. Web Service Modelling Language (WSML), http://www.w3.org/Submission/2005/SUBM-WSML-20050603/
6. Kenny, F.: Ahh Shucks, SOA Is A Failure (November 12, 2008), http://blogs.gartner.com/frank_kenney/2008/11/12/ahh-shucks-soa-is-a-failure/
7. InfoQ, Survey: Only 37% of enterprises achieve positive ROI with SOA, http://www.infoq.com/news/2007/08/SOAROI
8. Lawson, L.: IT BusinessEdge, Savings from Reusing Services Oversold and Overpromised, http://www.itbusinessedge.com/cm/blogs/lawson/savings-from-reusing-services-oversold-and-overpromised/?cs=33689&utm_source=itbe&utm_medium=email&utm_campaign=MII&nr=MII
9. Semantics of Business Vocabulary and Business Rules (SBVR), http://www.omg.org/cgi-bin/doc?dtc/2006-08-05
10. Halpin, T., Morgan, T.: Information Modeling and Relational Databases, 2nd edn. Morgan Kaufmann Publishers, an imprint of Elsevier (2008) ISBN: 978-0-12-373568-3
11. Nijssen, G.M., Halpin, T.A.: Conceptual Schema and Relational Database Design. Prentice Hall, Victoria (1989)
12. Stollberg, M.: SWS tutorial - Industry Workshop, "Adaptive SOA" Semantic Web Services-Realisierung der SOA Vision mit semantischen Technologien (February 20, 2007), http://www.wsmo.org/TR/d17/industryTraining/SWS-tutorial-potsdam-20070220.pdf
13. International Standard ISO IEC 9075:1999. Database Language SQL. International Standards Organization, Geneva (1999)
14. CA ERWin Data Modeller, http://www.ca.com/us/data-modeling.aspx
15. IDEF1X, USFIPS Publication 184 release of IDEF1X by the Computer Systems Laboratory of the National Institute of Standards and Technology (NIST), December 21 (1993)
16. ISO/IEC WD 19763-7, Information Technology – Metamodel Framework for Interoperability (MFI): Metamodel for service registration (Working Draft1, 2008-11-18)
17. Lemmens, I.M.C., Nijssen, M., Nijssen, S.: A NIAM2007 Conceptual Analysis of the ISO and OMG MOF Four Layer Metadata Architectures. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM-WS 2007, Part I. LNCS, vol. 4805, pp. 613–623. Springer, Heidelberg (2007)
18. Piprani, B.: Towards a Common Platform to Support Business Processes, Services and Semantics. Keynote speech at Open Forum for Metadata Registries, Seoul, Korea (June 2009), http://metadata-standards.org/metadata-stds/Document-library/Documents-by-number/WG2-N1251-N1300/WG2_N1259_BabaPipraniPresentation6.pdf