

The SOA Magazine

Feature Article



Articulating SOA in the Cloud

by Paul S Prueitt

Published: December 3, 2009 (SOA Magazine Issue XXXIV: November 2009)

[Download PDF](#)

[Digg This](#) • [De.licio.us](#) • [Slashdot](#) • [Technorati](#) • [StumbleUpon](#) • [Google Bookmark](#)

Introduction

A Web service is a type of information exchange between business entities. It is now most often articulated using a standard or set of standards following best practices. There is some evolution and increase in social value possible. After a standardization process has matured, a single set of foundational concepts is available to be applied to the development and use of information exchanges. These foundational concepts may identify an optimal means to communicate between parties. Service contracts are then possible where parties are bound together within an architectural framework for a well-specified period of time. Once articulated in this way, Web services become part of a SOA (Service Oriented Architecture) information system. The devil is in the detail however; as past implementations have too often been not platform agnostic. The reasons for non-interoperability may include the consequences stemming from pure utility functions that optimize competitive advantage but which introduce non-documented features into .NET or J2EE deployments. Implementation has too often depended on how Web services will be processed with specific software in mind. This implementation problem is being overcome using SOA design principles.

Once SOA design is mature, one may articulate services with respect to the actions of a computer processor, a many core computing system, or grid of computer processors. This "articulation towards the processor" opens a new vista and may follow an evolution similar to how standards developed for information exchanges between businesses. If we call the SOA standards an "outward" articulation, then the inward articulation is this articulation towards the processor. Specifically, SOA service specification, use, reuse and orchestration may be "double articulated" by adding the inward articulation, and thus become part of a computing backplate [REF-1]. The goal of SOA with a backplate is to simplify best practices used in all Web services, service contracts and service discovery/orchestration in a platform/vendor agnostic fashion. Discovery and orchestration is seen a part of a set of standardized mechanisms, including some for security and some for optimization of computer processor tasks. Using a simple metaphor, the standardized parts of services, contracts and mechanisms have been developed over time. These basic elements then form the support for any "outer" interaction between objects or Web services. To create the inner articulation this interaction is then analyzed with many core and grid computing in mind. The architecture that supports this inner process orientation has capabilities that existing architecture does not have. One of these is provable interoperability and a high level of transaction security.

Articulating SOA in the Cloud

Double articulated SOA service specification, use, reuse and orchestration will simplify best practices used in all Web services, service contracts and service discovery/orchestration in a platform agnostic fashion. In many cases, cloud computing bypasses the hardware platforms through a process similar to single processor program compilation. However, the process starts with mature design and this mature design is guided by SOA design principles and standards. A service is first articulated using SOA design principles, contracts and concepts. If all services within an infrastructure share a common second articulation, an inner

articulation, then we have double articulated SOA infrastructure. This second articulation is to a small set of binary footprints. It is inner because the articulation is to a small set of process elements each having methods that assist the available processors in task management.

The transaction architecture that supports inner articulation has capabilities that existing SOA architecture does not have. These capabilities include very high measures of information security and an intellectual property protection regime based on superDistribution [REF-2]. Because of an enforced regularity in computing orientation, double articulated specification would, in theory, create device independent secure markets for digital properties such as video and audio products. This is because a reduction of complicatedness in processor agnostic processor tasking allows several new types of encryption and compression. Compression technology may see a one or two order of magnitude efficiency while automatic mapping of all core processing elements to any available many-processor capacity.

Simplification to the Optimal

Current SOA certification materials may be seen as a "simplification to the optimal". The simplification occurs as bypasses are discovered to common previously hidden difficulties. This simplification phenomenon occurs when something is discovered and then agreed to as a standard. Because simplification to the optimal is an essential concept required to understand double articulation, we will give a simple illustrative example from the history of mathematics.

Up to a certain time in history, the square root of the negative number was just a conceptual black hole. No one could "go there". However once one adopts a single convention; that of calling square root of negative one the symbol "i", the understanding of a large part of what would become classical wave mechanics is made available. The adoption simplifies a collective understanding that emerged over time about wave mechanics. Simplification to the optimal may be contrasted with random innovation designed to create competitive advantage in a marketplace. Instances of random selection may be the selection of DOS as the monopolized standard disk operating system, the adoption of RDF (Resource Description Framework) essentially driven by DARPA and W3C as a standard knowledge representation in the Semantic Web. These selections were innovations designed to limit access, and thus to gain an advantage, to desktop computing or to human knowledge representation tools. These selections will hold back the advance of information systems until a bypass is discovered and a new standard is adopted.

In the case of the global SOA community, the methodologies, programming languages and business paradigms are finally become platform independent. It has been a long journey. We now are fully informed of the range of difficulties that always occur when systems are not properly prepared for, designed and implemented. In SOA certification materials these facts are strongly ordered by a theory of organization that is natural and complete. The materials have become simple and optimal, and not creating competitive advantages for one group over another group. The articulation of services and service contracts is thus made via a simplification allowing common clarity and deep efficiencies.

In upcoming books, implementation best practices are developed separately for .NET [REF-3], Java [REF-4], and REST [REF-5]. These implementation practices are based on the unique and often undocumented ways in which the first and second of these two paradigms have handled everything all along this long standards history. How are SOAP messages to be handled? How are WSDL documents to be handed? We find common cause of failures in projects that thought or acted as if .NET and Java interoperability would be seamless. Project failures occur because of the undocumented features build in to not allow data interoperability. These project failures hold back the development of cloud computing demonstrations. The up coming books will reveal these hidden features and show how to overcome them.

What cloud computing finally does is to say, hey we really need as a society to not care about hardware platform, delivery device or programming language. The seams created by an overly competitive spirit need not be there. When these seams are not there we find RESTful SOA, as an extreme simplification of the compounded and non-simple standards from the past. It is predicted that RESTful SOA will simplify service and contract specification by bypassing the un-documented seams that make .NET SOA and Java SOA not function together. As this happens, the entire SOA paradigm is finally seen as platform independent.

How to articulate to the binary is then the possible next step. This is the inner articulation of a system that has a double articulation. The key to this second articulation is in seeing measurement instances as parts of normal distributions. The reason why has to do with the nature of the physical world and the nature of

category, prototype and instance. Failure to see measurement of an instance placed into a normal distribution a measurement is due either (1) incompleteness of the set of distributions and/or (2) the presence of novel elements to the object of measurement. Seen in this way, the normal distribution is merely a conceptual container for a set. This set is the prototype and the set of instances that are the same as the prototype except for small-allowed variations from the prototype. The inner articulation is possible because in each case, an instance is a composition of a set of prototypes. Given some technical background, cluster analysis is shown to identify the set of instances from which a normal distribution will occur if the prototype is well selected. A few other technical discussions lead us to see that instance fitting into a composition of prototypes is not so difficult when, and only when, the set of prototypes are complete, have been optimized through simplification, are as independent of each other as possible, and if the measurement system has the ability to recognize actually novelty when it occurs. In the case that novel is being recognized, a set of mechanisms must be available to address this novelty.

Simplification to Optimal is a Methodology

Simplification to optimal is a methodology that will produce a class of discoveries all of which share the following observation. Real power comes when simplification is along the line of a real constraint. For example, when we by-pass the non-interoperability seams placed into standards by competitive corporations we simplify the design effort. When cloud computing bypasses these seams it must do so by using a simplifying discovery. It, cloud computing, has to be simpler and more powerful than the dysfunction offered by corporate IT.

Another example of simplification to the optimal is the use of the positional notation and an appropriate number base to represent a string as if it were a number [REF-6]. This number/string then has a determined place within a natural order, and as a consequence an order exists without having to be imposed. The use of positional notation, to index all information, is possible when one selects a number base having more symbols than characters in the alphabet. When this occurs one has a data persistence paradigm that is simply and more powerful than XML or relational database management systems. A few additional simple perceptions are needed to replace relational databases and XML. Data persistence and representational state transfer are then both enabled without any server technology. Computer servers then become not as important, as individual transactions have become agnostic as to the hardware. A few nice steps in elementary number theory and encryption theory allows this persistence and transfer paradigm to occur in a secure fashion, and one additional discovery allows one to not have to move data physically, and yet have the data grown at various locations. This data move in location follows the work on fractal image compression, but is also simpler than fractal image compression.

As suggested in the previous paragraph, double articulation is based on, separate and non-removable, discoveries in five or six other forms. In each case, a simple perception is made that is counter to how computer theory or information theory was set up. The foundational elements supporting the current paradigm are each being bypassed. To understand each of the discoveries some history is useful. We recall that computing theory has developed as if the early work was the best work and that new work must extend the truth of the former work. This seems reasonable, at first; but looking back we see that things have been set up to reflect the utility of market competitiveness, and not social value. Over time IT has become too expensive and deemed not responsive to social needs, for example in the health care sector or the education sector [REF-7]. So we develop computing theory based on certain types of mythology, like the AI mythology, not robust analysis based on systems theory.

The loose paradigm developed to support the current IT sector is not acceptable in the natural sciences where we need consensus on elementary notions, like what is "information". Because of this weak development philosophy, we built into computer science some deep flaws. Any time one is able to remove one of these flaws, one will have exposed a special discovery. These are special because each of these flaws hid a separate constraint on communication mediums.

The principle that optimality may be discovered via simplification has grounds in both formal mathematics and in natural science. In my upcoming book, *Bridge to the Future*, I have claimed that the class of these special discoveries cannot be fully contextualized without neuroscience, a clear understanding of biological mechanism, and the physics of emergence. In particular generic models of replicator mechanisms in biological systems show a dependency of any physical emergence on the real natures of non-local causation. These are big issues for science, always have been. However, these issues are being resolved, and with

this resolution it is possible to create a "special technology"; one that is double articulated and open to the measurement of novelty. I have pointed out that SOA is about articulating the behavior of services. REST goes in the direction of an articulation of representational states, in the context of how these state might be "moved" in the cloud. So RESTful SOA is double articulated. My initial use of double articulation is not; however, based on REST (REpresentational State Transfer) [REF-8] but on general framework theory [REF-9]. We have some work to do to make these distinctions clear.

So What is Next?

Every one of the flaws in current information science is due to its artificial nature, a nature falsely celebrated. Double articulation does link physics to human perception, but it does this through a comprehension of physics and human perception. The brain works because it uses the physical reality in a specific way. The backplate in physical reality is a set of universal laws; the laws of physics. These laws structure the expression of the stuff of the world. The classical laws are not sufficient, however. Penrose [REF-10], for example makes this in-sufficiency argument. He in quantum mechanics, as does Whitten in string theory [REF-11], shows that three complete theories of everything exist. These theories are now commonly seen to be perfect in all but two respects. The first is that the theories are absolutely independent of each other and thus create to problem of deciding which theory is correct. The second is that none of these theories tells us any essential thing about what life is, or how human will is expressed. They are deterministic models. Somehow the indeterminate nature of the real world must be reflected in how computing serves humanity. This leads to the conclusion that the next step is a computing backplate, having formal properties of formal closure and the mechanism allowing this formal closure to be modified in real time locally in the presence of measured novelty. Further mechanisms must then allow, or not, for the modifications to propagate within the cloud.

With such a capability, each person has the functional ability to create an innovation within the cloud process. The demand of the individual is then consequent to changes depending on mechanism within the backplate. The conceptual difficulty to this shift in information technology is evidenced by its history. Demand side implies that there is no help desk. There is no place to sale something. All design is done in real time as individual humans communicate. The interaction dynamics is Nash like in nature. The demand is from individuals, and the computing backplate has as its single purpose the continual maintenance of signs/symbols sufficient to the purposes imposed by humans in a communication medium. Like with natural language, but now the medium is cloud computing.

How to Open a Formally Closed System

The term "clopen" was first used in the context of computing theory in discussion between Sandy Klausner and myself in 2005-2006. My training in mathematical topology and axiomatic allowed me to see the usage. Most computer scientists do not have this training. Klausner saw a requirement that a formally closed system needs to be opened and then re-closed in a particular fashion. At first, the concept of clopen-ness was a hard one to get his mind around fully. Once he had this, he was off and running, using his concept of closure to articulate any new design process using only six-sided framework generated sets of articulated binary footprints. This was a early "inner" articulation.

The closure of a design was then a property of a designed process that would act as if compiled in any environment including many core and grid computing environments. The exposed flaw in computer science is the assumption that no permanence could be found out of the invariance of the output of a "global" compiler. The clopen mechanism bypasses this flaw. Instances are chunked into categories. The use of a high level chunking of data and process templates is the brilliance of Klausner's work. The formation of generative, sub-structural frameworks, and a design language that uses these and only these chunks is an additional innovation. The generative framework creates commonalities that are sharable within communities.

The set of basic elements required to run the generative process was closed and included only a subset of the set of all basic elements adopted within an infrastructure. Designed processes become articulated into these chunks. The chunks form a closed world that is then usable by processor task schedulers. A system is closed if every desired state is reachable. The clopen property allows formal means to address conditions where a desired state is unreachable. A closed system may be opened to reach a new state under special

circumstances. Once the system is extended the system must be shown to the closed. Design closure is required at design time. So these are the issues we dealt with.

SOA articulation and design articulation of data footprints might now be combined with mechanisms for running these processes in any kind of cloud environment, and thus to be cloud neutral. There is a lot of theory here so I will be brief. Any one of these small set of footprints is a model of a normal distribution of possible data standards definable using WSDL (Web service Description Language). In theory, and in practice, any time one uses a sufficiently delineated set of independent categories then one finds normal distributions of occurrences with real data. The prediction that a field of normal distributions will always exists has to do with two things (1) reification of universals from the instances of particulars and (2) the presence of replication mechanisms. One can take as a given that nature is as it is because of the presence of replication mechanisms. One may also observe that the production of universals is part of the human induction of cognition. Thus the inner articulation task seems simple. Find a sufficiently delineated set of independent categories so that these normal distributions may be measured in real time. This is what the immune systems, plants or animals, do. The degree to which this sufficient delineation has occurred may be measured by the emerging clarity of enumerated frameworks, such as what Klausner has produced.

We are not finished with this work. A methodology is needed to guide cloud computing process design in institutions having (1) profound legacy IT issues (messy and fractured databases, data design problems, data non-interoperability) (2) profound level of confusion and frustration on the part of system's users and designers (communication problems, personality and ego issues that arise all too often, cross purposes from different stakeholder communities). In these cases, the emerging SOA in the Cloud best practices go directly at remediation and enhancing existing systems, no matter how poor they operate now.

References

- [REF-1] Prueitt, Paul (2009) - "The Service Engine: Structured Communication using Modern Service Technologies" SOA Magazine, <http://www.soamag.com/I30/0709-1.php>
- [REF-2] Ryoichi Mori, Masaji Kawahara, "Superdistribution: The Concept and the Architecture". Transactions of The Institute of Electronics, Information, and Communication Engineers, vol. E73 #7, July 1990, pp.1133-1146.
- [REF-3] Erl, Thomas (in progress) "SOA with .NET & Azure"
- [REF-4] Erl, Thomas (in progress) "SOA with Java"
- [REF-5] Erl, Thomas (in progress) "SOA with REST"
- [REF-6] This discovery was patented in 1999, but is of such an extreme power that one may make a claim that no ownership may exist . In any case, a generalization from this patent has occurred in the form of elementary category theory.
- [REF-7] Prueitt, Paul S (under review). Bridge to the Future, a proposal to President Obama
- [REF-8] Fielding, Roy Thomas (2000) Architectural Styles and the Design of Network-based Software Architectures (PhD Thesis University of California, Irvine)
- [REF-9] Prueitt, Paul S (2002) "Generalized Framework Theory" published only on the web <http://www.ontologystream.com/beads/frameworks/generalFrameworks.htm>
- [REF-10] Penrose, Roger (2004) The Road to Reality: A Complete Guide to the Laws of the Universe (2004, ISBN 0-224-04447-8 (hardcover), ISBN 0-09-944068-7 (paperback))
- [REF-11] Whitten, Edward (1996) Reflections on the Fate of Spacetime, in Physics Today April 1996 <http://www.sns.ias.edu/~witten/papers/Reflections.pdf>

THE PRENTICE HALL SERVICE-ORIENTED COMPUTING SERIES FROM THOMAS ERL

