Editor: Charles Petrie • petrie@nrc.stanford.edu

No Science without Semantics

Charles Petrie • Stanford University

Recently, I argued with a colleague that even though he was a university professor, he wasn't doing science — at least in one case. He was offended. But I had to persist because rather than sharing his complete results with the academic community, he was hiding them as proprietary technology inside a company. This is a common practice, and indeed his country's intellectual property laws practically force professors to protect their work by moving research results to private companies. There's nothing wrong with this, and he might be a perfectly good scientist in other respects, but in this case, it's just not science.

Thank You for Asking

Doing science means using the scientific method. That includes reporting results so that others can repeat and verify them. Without this step, we're back to alchemy, and who wants to publish alchemy?

Much of computer science faces the criticism that it isn't science. That has certainly been a critique of that part of artificial intelligence (AI) that wasn't well-founded in a logical formalism, and was a source of tension between the "scruffies" (who typically produce results as programs motivated by intuitions) and the "neats" (who typically produce results in logic with proofs). Yet, that particular holy war is more of a red herring. If scruffies describe what they're doing well enough to let others replicate it — even if the work isn't built on some logic with formal properties — it's still science. But when a researcher reports phenomenal results that can't be replicated (you know who you are), we sadly have to categorize the author as a brilliant alchemist. And this diminishes what we do as a science.

This aspect of science has semantic implications. Given that other people must precisely understand the experiment and the results to be replicated, the semantics of the terms used to describe the experiment must be precise and consensual. If we don't use common terms with precise semantics, we might as well encrypt the results. Yet, the problem might be with the community as a whole, rather than any individual.

In the AI subareas of software agents and, now, Semantic Web services, this lack of scientific method has created confusion and a lack of clear results, and it's impossible to fix without common definitions of the fundamental terms. Furthermore, these definitions must distinguish the designation from other terms. If we claim that *foobar* is a wonderful new technology, aside from defining its wonder, we need to be able to say how foobar is distinguished from the previously well-understood technology *rebar*. And the definition can't be subjective. It does no good to claim that something is foobar just because we feel that it's so.

In software agents, there was never a definition of "agent" that was operational: a clear repeatable test on which everyone could agree with minimal subjectivity. The most common definitions that appeared in papers used terms like "autonomous" and "intelligent" with no way to determine what those words meant: "agent" was in the eye of the beholder.

Disclaimer: This Has Long Been My Pet Peeve

In 1996, I published at least the outline of an arguably operational definition of an "Internet software agent" that was widely ignored.[1] My primary concern is that the lack of fundamental definitions makes the work unscientific. When researchers claim to obtain a certain result because an agent has "beliefs" and "intent," the result isn't intelligible much less repeatable unless these properties are precisely defined. Moreover, the community misses a chance to perform deep analysis of

continued on p. 94

continued from p. 96

what the researchers are doing.

My 1996 nascent definition included the issue of whether a message was a reply to a previous message or a "surprise" message volunteered by a candidate agent. Thinking about this over the years, I've realized that this isn't a simple determination; indeed, it's at the heart of what we consider to be intelligence. But there can be no community analysis of this or related issues when everyone is content to claim agenthood because they consider that their agent "senses that environment and acts on it"[2] — as if this distinguished agents from print servers.

Developers shouldn't be able to mean such different things that incompatibilities become evident only at build time.

Finally, this lack of definition leads to confusion inside the community because many people are using the same words with different meanings. I mentioned one example of such confusion among fellow agent developers in my first Peering column, "Pragmatic Semantic Unification."[3] Developers shouldn't be able to mean such different things by "exchanging agent messages" that incompatibilities among systems become evident only at build time. This isn't even engineering, much less science (ignoring for this column that I might distinguish those terms, as you can see all the deep rabbit holes into which we might then fall).

Christoph Bussler and I examined this confusion concern with respect to the Semantic Web service community in a column entitled, "Industrial Semantics and Magic."[4] Such confusion is an important issue and will naturally occur when the fundamental terms aren't well defined. I'll now stop for a moment and allow you to review that article.

Pause.

OK, you didn't read it, but here's the bottom line: we simply can't depend on Humpty Dumpty's definition of semantics from *Alice in Wonderland*. ("When I use a word, [...] it means just what I choose it to mean — neither more nor less.") How can we be doing science if we don't agree on a definition of *Web services*, much less *service composition*? Even more embarrassing, how can we claim to be doing "semantics" as a science when we have no semantics for our fundamental terms?

For a definition of Web services, I refer you to one developed at a Dagstuhl workshop (http://drops.dagstuhl.de/opus/frontdoor.php?source_opus=526) that distinguishes this technology from remote procedure calls and a general notion of services. When I see articles about, say, how the telecommunications industry has really been offering such services since before the Internet because it offered services with well-defined APIs, I believe the authors have missed the point about the feature that distinguishes this new technology: the publication of the service descriptions in a standard machine-parsable language on the Internet (usually the Web).

Were the authors to carefully offer a new definition that described the prior services as a superclass of Web services while preserving a definition of Web services that distinguished it as a technology, then that would be helpful for communication; using an abstraction to claim that there's no important difference is not.

When I see articles about "Web service composition" that describe solving very different problems, I also see a need for more precise definitions. When I see articles that purport to compose services using "planning" but omit from the problem domain actions that change state in the world, it's clear that the authors are misleading the reader, and perhaps themselves.

The problem occurs even in the very notion of an *ontology*. For many computer scientists of the formal ilk, a *vocabulary* is a set of terms used in inferencing, a *taxonomy* is a simple class structure of terms, and an ontology requires stronger restrictions on the use of its terms by logical axioms, preferably as a machine-consumable computational logic. Yet, for some people working in the area of representation, all of these are ontologies. Certainly the computer information systems community considers the Resources-Events-Agents model (www.msu.edu/user/mccarth4/rea-ontology/) to be an ontology, despite the fact that it isn't described by a formal logic (although it seems very likely to be in the near future).[5]

A Grand Challenge

This definition problem certainly occurs outside of the areas I've discussed. For example, I work on a Darpa project called Transfer Learning (www.darpa.mil/ipto/programs/tl/) that suffers from it as well. Several groups across the country are developing technology for it, and yet the project has no formal definition of what "transfer learning" actually is. Although it might seem intuitive to understand that we could learn something from checkers that applies to chess, trying to formalize this notion in terms of problem solving and search is extremely problematic. So how do we know what the learning agent developers are even claiming in a scientific sense?

Finding consensus on formal definitions for intuitive notions is hard but worthwhile. Alan Turing's notion of a computable function is the exemplar. An important aspect of such definitions is that they be clear and crisp, but such clarity is difficult to achieve. We all spend our days, starting as students, learning to write more clearly. Indeed, I tell my students it's more important than being right: if you're clear, you'll find out if you're right much sooner. And, of course, finding such clarity is the main difficulty in the early stages of a scientific investigation.

Science often starts out by fishing in muddy waters. We can trot out some definitions and see if they work, but if the exercise of discussing them widely isn't part of the technical culture, we'll make no progress. It seems that astronomers are more interested in the definition of a *planet* than we are in definitions that might make computational differences and define our science.

Fortunately, there is another, more popular, way to achieve consensus on what we mean by technical terms, assuming that we can actually find important technical differences to designate: challenges.

In the Transfer Learning project, we jointly design a set of problems and a testing protocol. This is difficult without a definition of what we're testing, but if we follow our intuitions and analyze the consequences carefully, we develop (and have) a testing protocol that de facto defines what transfer learning is.

Can people build agents that are smart enough to respond well to new problems? What does that even mean? Our group offers the General Game Playing Challenge (now in its third year at the AAAI Conference on Artificial Intelligence; http://games.stanford.edu/competition.html) as a test. Can your agent read the rules of a game it has never seen before and then play it well? Come prove it. Can your Web service composition agent solve problems? What kind of problems? How well, and with what help from a human programmer? Come find out at the Semantic Web Services Challenge (www.sws-challenge.org).

My bottom line is to appeal to all of us to do science. We can experiment and not be sure of what we're doing at first. But eventually, we, as a scientific community, need to be able to determine whether a result is cold fusion or the bomb. And we can do that only if we know what we're talking about. Without consensus semantics, there can be no science.

Reference

1. C. Petrie, "Agent-Based Engineering, the Web, and Intelligence," *IEEE Expert*, vol. 11, no. 6, 1996, pp. 24–29; http://www-cdr.stanford.edu/NextLink/Expert.html.
2. S. Franklin and A. Graesser, "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents," *Proc. 3rd Int'l Workshop on Agent Theories, Architectures, and Languages*, LNAI 1193, Springer-Verlag, 1996, pp. 21–35; www.msci.memphis.edu/~franklin/AgentProg.html.
3. C. Petrie, "Pragmatic Semantic Unification," *IEEE Internet Computing*, vol. 9, no. 5, 2005, pp. 96, c3.
4. C. Petrie and C. Bussler, "Industrial Semantics and Magic," *IEEE Internet Computing*, vol. 10, no. 4, 2006, pp. 94–96.
5. F. Gailly and G. Poels, "Towards an OWL-Formalization of the Resource Event Agent Business Domain Ontology," *Proc. 10th Int'l Conf. Business Information Systems*, LNCS 4439, W. Abramaowicz, ed., Springer, 2007, pp. 245–259; http://users.ugent.be/~fgailly/phd/docs/OPSW.pdf.

Charles Petrie is a senior research scientist and consulting associate professor at Stanford University. His research interests include concurrent engineering, virtual enterprise management, and collective work. Petrie has a PhD in computer science from the University of Texas at Austin. He is EIC emeritus and a member of *IC*'s editorial board. Contact him at petrie@stanford.edu.