The Rubicon of Smart Data

Roger (Buzz) King

Computer Science Department
University of Colorado
Boulder CO 80303
USA
roger@cs.colorado.edu

Julius Caesar crossed the Rubicon River in 49 B.C., thereby irrevocably committing the Roman Empire to war.

We, as researchers have repeatedly promised that soon, individuals will be able to easily locate web data according to precise semantic specifications, retrieve that data, have it automatically integrated, and then have it presented to us or our programs in a compact, highly usable format. Gone will be the days when a user invokes a generic search engine, receives landfills of URL's, chases down these URL's and screens them for content, extracts whatever is useful from the resulting web pages, and then painstakingly integrates this information and prepares it for processing.

Each person, group, or program will have its own information space that is always quietly evolving behind the scenes and according to the owner's wishes. These information spaces will be simple to share, thus allowing us to easily trade data and build new information spaces out of old ones. Information spaces will even become predictive by learning owners' data habits, and will thus offer up highly valuable data that wasn't even requested.

Some significant results have been produced already. This is good, of course. But these tantalizing software tools, combined with our repeated promises of vastly more powerful tools in the near future, have forced us to cross the Rubicon of Smart Data. We no longer have the choice of turning back and still saving face. We must deliver the "semantic web". As database folks, this means that we must solve a long-standing problem that lies at the heart of all forms of smart data. We must find a way to capture the semantics or the "meaning" of data.

**Proceedings of the 28th VLDB Conference,
Hong Kong, China, 2002**

Julius Caesar won his war. Will we win ours?

**

In the future, structured, standard terminologies will be used to annotate data. We are so confident of their immense power to capture semantics far in excess of those captured by traditional metadata, that we have given them the lofty (and silly) name of "ontologies"*. Many industrial and research groups are currently involved in terminology development efforts. Several substantive terminology taxonomies exist or are under development. PubMed (http://www.ncbi.nlm.nih.gov/entrez/query.fcgi) is one example. (Also look at http://www.ontology.org .) Many ontologies will detail not only the static structure of data, but also the main operations used to create and manipulate it.

Various software levels will leverage the utility of these ontologies. Mediators (AKA smart wrappers), will be fed web data, interpret it via these terminologies, and reform it into something - if you believe the marketing slant of the semantic web research world - perfectly fitting the occasion.

There are already a number of commercial systems that can be used to integrate heterogeneous databases as well as various forms of web data; one such product is Cerebellum (http://www.cerebellumsoft.com). For the most part, these products, use the relational model as a way of representing the common form of data and the operators that are used to integrate data.

The software layering and the promises go on. Agents, armed with user profiles and declarative user requests, will find just the right stuff, and then use mediators to extract, integrate, and reformat data. Agents will often animate data, making it sing and dance on our

* Indeed, an ontological argument is one that has to do with the meaning or reality of existence, and thus, our use of the word is somewhat ironic – structured terminologies are by their very nature artificial and not directly reflective of any true existence.

screens. One limited, but very intriguing agent product is Agent Sheets (http://www.agentsheets.com).

Underlying and overlaying all of these existing and soon-to-exist products will be a host of tools (making up a host of other layers) that will support the identification of Internet resources in a location-independent fashion, the display of data and resources in highly adaptable ways, and the complete platform-independence of data. All of this, from the lowliest file system to the most sagacious of agent software, will be integrated, top to bottom, in one elegant view of data. A good place to poke around for the big view of things is http://www.w3.org/2001/sw .

And so, gone will be the problems presented by the heterogeneity of web data (images, sound, video, text, relational tables, etc.), the vast size of the Internet, the conflicting terminologies and data formats of various websites, the incompleteness and unreliable accuracy of web data, the difficulties of precisely identifying a user's needs and of locating relevant data, and most of all, the tendency for the semantics of data to be tied up in countless lines of code that manipulate data or in the minds of interactive users who must personally interpret that data.

Even the vast amount of hidden data, that stored in files and databases, and accessible via a website only by experts who are aware of its presence, will gradually become visible and usable by all of us. This will (of course) be accomplished by more software layers.

**

This is all rather prophetic and for some, hard to believe. But some things do seem intractable at first glance, but have reasonable, pragmatic solutions. So, how much of what is promised above is quite deliverable?

One ageless question lies at the center of this. How *do* you identify, extract, integrate, translate, and present the real "semantics" of data?

**

Data is by far the most prevalent human-created artifact, and we have always depended on the mental power of humans, either directly or as it is encoded in programs, to figure out the meaning of data. And we have always used one primary technology for assisting this process: metadata. All we can really do is sharply increase the effectiveness of metadata and to facilitate as much as possible the processing of metadata by programs.

In a sense then, we are collectively trying to deliver on an inherently intractable problem. Many "hands-on" computer professionals doing real jobs in the real world see us researchers as dreamers, or even as liars. When they see our relative failure at solving a huge world of problems, including this one, they often give the following argument: Computer "science" researchers have insisted on attacking problems as if they were mathematicians or true scientists, and have thus limited themselves to finding solutions that are either mathematically pure, or based on some natural model, or at least beautifully elegant while being fully functional.

We don't want to admit that the army that crossed the Rubicon of smart data is an army of blue-collar engineers. Our knuckles are skinned, our palms are calloused, and our clothes are spotted with grease, and the "smart" machines we build are feeble. Yet we proclaim ourselves scientists and mathematicians, and shrug off the many-layered software solutions as things that will disappear just as soon as the "real" solutions are discovered.

That's what they say about us.

**

But this is changing rapidly, and a new order of nonsense will soon reign. Computer "science" has become more and more a respectable realm of engineering, and that realm is quickly proving itself salable in the marketplace (unlike, the realm of "pure" but unrealistic solutions).

We used to imagine someday finding perfect, elegant solutions to intractable problems, while selling software solutions as temporary and nasty fixes; we now hide the core intractable problems inside layers of software and pretend that these problems don't exist anymore.

**

This is all part of a widespread phenomenon that can be seen throughout computer science. Consider the database world and the problem of extracting data semantics. The spectrum of elegant-to-engineering solutions is decades old and is based on the creation of metadata – schemas, annotations, structured terminologies, etc., and ways of manipulating that metadata. The elegant solutions started coming out in the seventies and eighties. When these proved ineffective, far less aggressive goals were laid out, and less functional but far more useful solutions emerged and have found their way into the commercial world.

So, as is often the case in computer research cycles, approaches to tough problems start out small-grained and semi-formal, and then expand hugely into very large-grained engineering techniques. We just don't want to admit that we gave up on those core tough problems.

It's time to step back and accept our core problem of extracting semantics as inherently ill defined and intractable. Perhaps it is not possible with current software technology to avoid big systems that are not very robust, are very complicated, and once built, are hard to understand. But we at least need to build ones that replace the mythical, hidden gem with something far more extensively researched by a community with a shared technical agenda.

Let's go back, consider that core problem with the eyes of experience, and see if there is some other, perhaps less exciting core problem that could indeed be directly solved. In this way, we might build a new "semantic" web that is rooted in solid, well understood, elegant, and this time, realistic technical solutions.

**

The overriding goal of all this would be to develop better understood software layers that will be less likely to

explode into chaos as a result of their incredibly fragile, tangled, and non-extensible states. The eventual gift of theoretically and formally inclined researchers should be engineering solutions that are organized, maintainable, and truly extensible.

**

Let's quickly look at the engineering solutions that have evolved in the database world and relate to semantics extraction. Maybe an analysis of them will inspire us.

Work in the area of schema integration and evolution dates back to the very early eighties; there was a spike in this work in the mid to late eighties, and it continued on through the nineties. Most of these approaches were based on fairly low level, but somewhat formally defined operators that mapped schemas to common models (often object-like), and at the same time, resolved terminology conflicts. Other operators, more specifically dedicated to evolution, added attributes, created subtypes, etc., schmeas. (Try poking around http://liinwww.ira.uka.de/bibliography/Database/Schemaevolution.html .)

There have been some solutions that pretty much meet that ultimate test of elegance-turned-engineering. They should serve as powerful examples for those who attack the old semantics problem. The calculus of SQL nicely isolates the statement of the properties of needed data. And of course, SQL and many other languages and systems make powerful use of metadata, in the forms of schemas and data structure definitions.

In fact, data modelling has, for decades, been a high-volume research domain within computer science, and a number of noteworthy solutions have evolved. We've developed enhanced domain facilities for relational databases, as well as mechanisms for encapsulating the behaviour of data and the separation of signatures from implementations.

But for every great contribution, there have been a thousand shameful results. For at least twenty-five years, data modelling has been the philosophical orphan of the database world. How many papers on the art of modelling have been generated – and laughed at?

**

For the sake of being constructive, let's stick with the good stuff. Consider constraint facilities. These provide ways of specifying application logic and associated implication methods.

Also consider the rapidly emerging and already-mentioned standard, structured vocabularies, some of take on a bit of a formal nature. The central problem with these is that there is a vast gulf between the tiny objects-attributes-axioms models they embody and the dramatically larger grained semantics that truly useful ontologies would need to embody in order to really make data "smart".

There are other very useful approaches that lie somewhere in the grey zone between elegant and sloppy, and the overall approach of putting information into databases is one of them. Database technology does a good job of capturing the underlying, long-term image of a business, along with the intended means for allowing the state of the image to change over time. But this is only a snapshot of the already-stilted forms-based approach to data processing that existed just before the great era of computerization began in the sixties. It's when we try to more deeply capture semantics that we fail. We find that the true meaning of data is tangled up in our brain cells and our lines of spaghetti code.

**

This last point is key to grasping the true depth of our semantics problem. It is still very difficult to know what a large system of programs "does", and what needs to be done to understand and evolve a large system that consists of programs, transient data, and persistent data. And we are relatively impotent when it comes to reusing data and programs in anything other than their original fashion.

This is why websites are focused largely on multimedia display, support very rigid interactive capabilities, do not lend themselves to machine processing, and tend to conceal the true forms of data that underlie their visual displays.

We need to clearly define our problem – and that is to quit trying to simulate human interpretation of data and cleanly define what we see as the reasonable limits of machine interpretation of data.

**

Let's dash out some relatively obvious approaches to finding these limits. This might inspire us even further.

We could imagine incrementally developing very complex extensions to natural languages such as English. These would allow us, as a global culture, to agree to standard ways of referring to business, academic, scientific, engineering, etc., data and corresponding processes. Then, rigid, unambiguous subsets of these extensions could be used as the basis for capturing semantics.

In a given realm, there could thus be two structured vocabularies, one dedicated to the structure of the given domain, and a vastly larger one that is shared by all of computerized society.

While we're at it, how about a third ontology, one that provides a common way to define additions to web-resident hypermedia, so that users and programs could selectively enrich the material? These would form standard categories, such as reference links (which might well be further annotated), images, related textual material, etc. This conjures up a new, vast suite of processes, the ones that are involved, not in the original creation of data, but in its extensions and reintegrations. We would in a sense have a global ontology that trains all computer literate people in the skills of data reuse.

. Hmm. Sounds like a super mediator specification language. Might be intractable to build, huh?

**

Let's try to extract a core, approachable problem out of this muddle of mega-ontologies. Maybe we need to develop an understanding of what it means to extract and integrate data of many diverse forms. The current trend is toward using OIL/DAML, which provides a representational language for specifying classes, properties, etc., as well as an ability to specify axioms that serve as the basis for inferences. Such axiom-based languages provide not only a formal underpinning for declaratively capturing semantics, but also for merging ontologies. These formalisms can be quite tedious to use, and do present problems when used in a very large scale applications where the schema to data ratio is large.

In other words, they capture small grained semantics for business applications, and do not perform well when applied to larger grained semantics in engineering, scientific, and academic (such as historical) applications. There are thus two critical scale problems, the size of the semantic elements and the size of the metadata as a whole.

How can larger grained semantics be embedded in web data? Automated extraction on multimedia data is not a new idea. Image processing folks can isolate faces from security photos and runways from satellite photos. Text processing systems can search for words, formatting markings, and complex sentence structures, and then pull out pieces of text. There are many other examples, in particular in the engineering world where massive, complex designs must be analysed and visually displayed from many different angles. These solutions should be looked at as potential sources of generalization.

**

In sum, the emerging "semantic web" will not fully emerge unless we can turn large numbers of URL's into meaningful and compacted information that is targeted for a specific user or program. This will affect those trying to perform research on the web, develop new web retail and b-to-b businesses, and establish effective community web portals.

Large businesses and other organizations will not be able to take full advantage of their diverse, widespread, and content-overlapping databases until data can be located, extracted, and integrated semi-automatically. This will impact the development of new information-based products, the usability of data warehouses, and the consolidation of information-addicted corporations.

As a culture we will not be able to take full advantage of electronic data until we have developed some widespread, concrete understanding of how to capture the processes of extracting and integrating the semantics of wide forms of data. This understanding will have to be captured in software tools that are easily usable by interactive users and by of "smart" software.

There is a core problem that underlies all of this. Indeed, we see that the problem of capturing the meaning of data has all the earmarks of a grade A problem, the kind that's worth getting excited about:

- It is a long-standing problem that has popped up repeatedly over a period of many years, and in many different contexts.
- The solutions that have been developed so far are a mixture of partial, formal and semi-formal solutions, as well as multi-layered and complex, heuristic-driven software solutions.
- The problem has been carefully isolated by many researchers because it is of acknowledged general interest, and because without isolating it, surrounding problems are difficult to address.
- Finally, the problem cries out for redefinition, in order to isolate something that would be truly useful while still being tractable.

**

Such problems warrant focused attention by a research community. We might accelerate progress by working hard to keep the problem visible. There are concrete research steps that could be taken.

What can be abstracted from the many software solutions that have been developed? Can we look at the cores of these solutions and extract a redefined notion of semantics extraction?

These software solutions include schema evolution and integration facilities, tools for integrating multiple databases into conceptual or materialized wholes, SQL view mechanisms, tools for creating integrated warehouses, mediation software, multimedia systems, agent systems, machine learning techniques, website and portal design tools, and ontology/logic languages are to a large degree heterogeneous in their nature and functionally isolated from each other. All of these do some form of semantics extraction and/or integration.

It might be that the rare formal gem that would support graceful scaling to very large ontologies, as well as accommodate very complex semantics does indeed exist, and is hidden in the kaleidoscope of software solutions. One technique that might pay off is the organization of a small team, consisting of database, AI, and semantic web folks. Such a team would remain intact for at least a couple of years, and might be able to somehow balance the software, formal, and human sides of the problem. Perhaps the clumsy and unnatural division of this problem among researchers in multiple, rigidly defined sub-disciplines within computing has prevented us from weaving together a richer, more powerful, and yet simpler solution. Of course, researchers from diverse areas have come together multiple times to attack this problem and related ones, but a longstanding, dedicated approach is needed.

**