technologies in unobtrusive ways. Specifically, we can introduce entirely new classes of data-driven applications and integration strategies. When we can name our data and ask for it in application-friendly ways, we facilitate a level of exploration, business intelligence, and knowledge management that will make most analysts drool when they see it. The Simile Project,[†] a joint effort between the W3C and the MIT CSAIL group, has produced a tremendous body of work demonstrating these ideas and how much drool can actually be produced.

Consider the scenario of tracking the efficacy of various marketing strategies on website traffic and sales. We might need to pull information in from a spreadsheet, a database, and several log files or reports from web analytics software. Although tying these things together now is not exactly rocket science, it does require a nontrivial level of effort to find, request, convert, and republish the results. If we simply produce a spreadsheet summary and email it around, we effectively lose the ability to retrieve the results at some future point without searching our already clogged inboxes. Adopting a CMS or other document management system that we can link to will increase the amount of time necessary to produce the result. Whatever the frequency is for generating these reports, we will have to repeat the process every time.

In a resource-oriented architecture, we could simply address the source of each of the data elements and ask for them as JSON files so they could be easily consumed in a browser-based environment. The Exhibit project[‡] from Simile with a Timeline view[§] almost gives us this ability. Throw in a little bit of work to convert Excel spreadsheets to JSON objects, and we have a reusable environment that, when in place, would allow us to assemble and republish these marketing reports in a matter of seconds. Now consider that the same infrastructure could enable the ability to bring other forms of data together as easily for different types of analysis and reporting, and you begin to realize the value of a web of addressable data. These kinds of environments are emerging in the Enterprise; if your organization cannot tie its data together this easily, it should be able to do so.

Applied Resource-Oriented Architecture

Recently, I built a resource-oriented system on the rearchitecture work my company did for the Persistent URL (PURL) system. The original PURL[‖] implementation was done close to 15 years ago. It was a forked version of Apache 1.0, written in C and reflecting the state of the art at the time.[#] It has been a steady piece of Internet infrastructure since then, but it was showing its age and needed modernization, particularly to support the W3C TAG's 303 recommendation and higher volumes of use. Most of the data was accessible through web pages or ad hoc CGI-

[†] *http://simile.mit.edu*

[‡] *http://simile.mit.edu/exhibit*

[§] *http://simile.mit.edu/timeline*

[‖] *http://purl.org*

[#] This codebase formed the basis of the very successful TinyURL (*http://tinyurl.com*) service.