News
When Web 2.0 Becomes Web Uh-Oh
Greg Goth

The promise of cross-organizational computing and communications has long been a Holy Grail for network architects. From the dawn of the Arpanet to today’s deployments of service-oriented architectures (SOAs) and remotely hosted applications, wider reuse of standards-compliant software components has been a constant goal. The rise of social networks and Web 2.0 principles are the latest trends in reusing software on nonhierarchical architectures.

When these architectures work—when user identities are protected, when process security is ensured, and when there is consensus on both technical and contextual attributes of a given piece of software—the cross-organizational model has great potential.

However, when the needs of disparate communities are not aligned, the results can be embarrassing, at the very least.

The classic cautionary tale
A classic example of cross-organizational cross-purposes surfaced recently, as explained by eWeek executive news editor Michael Hickins (http://blogs.eweek.com/epiphany/content/machinations/the_trojan_social_opensource_dropdown.html). His wife, who was in charge of online donations for a large, international relief agency, had received a strongly worded message from the leader of a pro-Israeli group accusing her organization of anti-Semitism and threatening to spread the word among potential Jewish contributors. As it turned out, Hickins wrote, “we learned that the ‘Country’ dropdown menu on one of her organization’s donations pages omits Israel as a country and includes ‘Palestine.’ Among other things, this means that Israelis can’t donate to the organization from these pages.”

The problem, Hickins said, came from the new cross-organizational computing infrastructure of many small commercial Web sites. The offending site wasn’t Hickins’s wife’s agency’s own Web site, but the site of Causes (www.causes.com/pages/tos), which puts widgets on social networking sites and lets those sites’ users donate to their favorite charities without having to navigate directly to the originating charity. Causes, in turn, uses nonprofit payment processing firm Network for Good for its transactions and also used (at least in the case of Hickins’s example) open source code from Ruby on Rails (RoR) to create its drop-down list of countries in which potential donors live.

Each organization appears to have made independent decisions regarding what nation or territory to include or not include. Although the technical aspects of the federated architecture worked well, the cultural and political contexts obviously created a problem. Specifically, the RoR organization included both Israel and Palestine (ISO and RoR classify it as “Palestinian Territories, Occupied”) in the list, but Causes, citing high degrees of Internet fraud in Israel, doesn’t allow Israeli donors to use its conduits to US-based agencies (http://apps.facebook.com/causes/help). To someone unfamiliar with the different motivations behind including or not including code in the end-user facing application, the decision might appear to be a political statement.
Hickins concluded, "I’m not bringing this up in order to cast aspersions on Causes or Network for Good—I have no reason to believe they are part of an anti-Zionist/pro-Palestinian conspiracy—but to point out how easy it is to lose control of your own self on the Web."

**Brands, standards, and the Web**

Causes developer Project Agape didn’t respond to an interview request. However, Michael Koziarski, the RoR developer who committed the default country code dropdown list in question, commented both on Hickins’s blog post and for *DS Online*. Koziarski says the RoR community expressly chose the ISO 3166 country code list to avoid political arguments rather than create them.

"Prior to our switching to the ISO list, we were constantly receiving patches and bug reports from people asking questions about their country," Koziarski says. "For example: 'It should be United Kingdom, not Great Britain, or 'Could you please make it Espana, not Spain?' and so on. Plus, there were cases of duplicates, such as Espana + Spain, UK + Great Britain."

After some discussion, RoR decided to sync their country list with something official. "The concern I had," Koziarski explained, "was that I wanted to avoid being drawn into political discussions similar to Red Hat (http://www.linux.com/articles/25749?tid+23, in which the Linux distributor got caught up in a row over removing Taiwan’s flag in its KDE [K Desktop Environment] files). RoR no longer takes patches that alter the ISO list, he said, "unless it’s to fix an error. Most of the community seemed very happy about this until the Causes incident."

Former Jupiter Research analyst Rob Sterling, now an independent interactive business strategist, says small and nonprofit agencies are most vulnerable to the unseen pitfalls of using code from other organizations.

"I think what happens sometimes is that small organizations adopt technologies not because they looked at all the options," Sterling says, "but because one particular option strikes them as interesting and useful. So they don’t shop around all the options they might pursue. Enterprise IT architects tend to work through several. They’re not doing a thing because it’s available, they’re doing it because they think there’s a business case for doing it.

"If you go into any large corporation, there are people responsible for defending the brand looking at this stuff. They’re going to have their own internal compliance rules that may be more stringent than federal or state rules. In a large corporation, you have to go through a lot of hoops in order to roll out a Web application, and maybe some of these smaller nonprofits should find a way to get some of that rigor before they roll things out."

Sterling says the potential embarrassment suffered by agencies caught in similar situations should encourage them to be willing to spend more money up front on user testing and software development and testing, if necessary.

"In this case it looks really bad, although everybody was acting with the best of intentions," he says. "Network for Good is handling fundraising for large numbers of these not-for-profits. The problem here is Network for Good was not the appropriate conduit for this charity. If you have anything that’s politically sensitive, you’re going to have to do your own online fundraising, all the way from the Web to the bank."

**Communities can learn from each other**

The promise of on-demand federated applications is piquing interest in sectors far beyond the budget-constrained nonprofit sector. Hub Vandervoort, CTO of Progress Software, says firms of all sizes are trying to figure out how to capture the best qualities of many-to-many Web 2.0 technologies while
preserving the integrity of traditional business relationships. Vandervoort calls this approach "socially-oriented architecture."

“How do I take this kiddieware and put it to use in a business context or Defense Department context?” Vandervoort says. “That’s what socially oriented architecture is about, and it roots in the basic premise that federation is the most powerful social structure for innovation. It’s better than distributed enterprises, it’s better than any centralized construction you can come up with. So how do you leverage that in the business context? Certainly, continuous delivery is one facet of that. But you also have to think about how to put together contracts and agreement structures in the absence of hierarchy.”

Sterling says the appeal of using free open source code should be leavened with the knowledge that some customization might be necessary.

“You have to budget for these things in advance,” he says. “If you write up requirements and have code written just for you, it will be exactly what you wanted. If you go to an open source project, it is meant to serve many masters. It is bulkier than if you used something written just for you. The drawback to that is that sometimes there are things in the core or plug-in modules that are not things you want. People need to check to make sure what’s going on.”

Certainly, organizations deploying applications shouldn’t expect an open source development community to write code that addresses specific non-technical issues, Koziarski says. He advises developers to be aware of potential controversial points for their target audiences. He also sees the Causes example as illustrative of what can happen once code is released into the wild. “It seems unlikely to me,” he noted, “that Causes actually was using our list, as they had ‘Palestine’ and the ISO list has ‘Palestinian Territories, Occupied.’ I suppose they copied the values then shortened them or something similar.”

Koziarski says he doesn’t believe it makes any difference whether a library is open source or commercial "or even internally developed" in terms of the risks of complications. “Whenever you take a piece of code, you need to understand the implications it will have on your application. Be it closed source, internal, or commercial, testing this stuff sounds like a good idea.”

Cite this article: