Deliver Now

Adrian Kosmaczewski

2007-11-11

Every time I talk with people about Ruby on Rails in Switzerland, I almost always get the same comments, no matter what is the background of the person I'm talking to:

> Yes but… what about [scalability / performance]? [I'm sure / I've read / I think / I believe / I have dreamt] that Ruby on Rails is not as [fast / scalable / powerful] than [J2EE / .NET / PHP / ASP / CGI / WebObjects / Python / Perl]

It's very funny indeed, for many reasons:

1. None of these comments came from people running something like Facebook, or at least any other site with more than 10000 visits per month;
2. None of the people who said something like the above has tried Ruby on Rails, beyond the 15 minute blog thing, which everyone seems to have done.

It seems to me that there's a problem here.

Twitter runs on Rails. And that's a heckuva lot of people posting updates every second. OK, this hasn't been a smooth ride at all, but Rails brings you productivity and maintainability, and this, more often than not, outweighs performance. The creators of Twitter did not know that they would be so wildly successful, in a way that has somehow redefined it as a new category of asynchronous communication. However, I think that if they had paid more attention about scalability from day zero, you know what? Maybe they wouldn't have **delivered Twitter at all**.

And you get customers, visibility and market position thanks to delivering products, not just by having a nice architecture.

And Twitter is not the only one: ManiaTV has more than **10 million visits a month** (source: HappyCodr). And Basecamp is used by thousands of users every day. Like Odeo, Shopify, Fluxiom or Strongspace. **Rails is ready for the real world. And it kicks your platform's a** every day.**

I know of a couple of rewrites in this precise moment: somewhere, out there in the wild, there is some team rewriting a PHP application in J2EE just because it doesn’t scale. Another rewriting a successful ASP application done in VBScript into ASP.NET and C# just because the code is a mess and CIO Magazine says that .NET is better. Another trying to rewrite a C++ application in a J2EE just because that’s how it should be done.

In my experience, rewriting is often just wishful thinking. None of these teams are ready to finish, none of these teams are ready to show anything to a potential customer. **They cannot deliver.** Deliver first, worry about performance later. I wish you luck and users and subscriptions and lots of worries about performance with all my heart!

Don’t get me wrong: I believe and always have believed that architecture is an important asset in the development process. However, when it gets in your way to deliver a product to your customer, it becomes a **liability**.

> Release early in the project and release often. Just like your application, your deployment doesn’t need to be perfect from day one. You can start simple and grow into more sophisticated deployment strategies over time.

James Duncan Davidson, read in Peter Marklund’s Rails 101.

By the way, Parking Friend in on HappyCodr too ;) The application is servicing around 100 customers per day after three weeks online (according to Google Analytics)… and for the moment it’s doing well! I’ve Globalized it recently, and also added credit card support with the ActiveMerchant plugin. All under 150 hours of work (around 19 days of full-time work).