Choosing a web framework for analysis

The current project is an open source data analysis platform. It's aimed at humanitarian and development data, but could be used to track other data too. It needs to work with existing big data and open data tools. And we have no budget to get it built.

The idea is pretty simple. Give each analyst access to data, tools and a place they can store “hunches” - ideas about what's going on in data that they haven't fully formed yet, and would appreciate some help from other analysts on. Make each platform capable of operating off-tether (for those unfortunate moments when the local internet is down). And connect the platforms together so that people with similar problems can share ideas, best practices, interesting leads etc.

Oh, and we have 6 months to build a pilot system, and are planning to use agile and test driven development techniques in the build.

Potentially suitable web applications framework candidates to carry this include:

- Ruby/Rails
- Python/Django
- Scala / Lift
- Java / Struts
- PHP / Drupal
- C# / ASP
- The results of the building a distributed decentralised internet group's work.

The desiderata for choosing a programming framework for this are:

- Help. How big and how healthy is its active community? i.e. how much help is available if we hit a snag or need to know about a feature?
- People. How easy is it to attract people with these skills onto the project?
- Privacy. How secure is the framework? Some data will need to be protected, i.e. not visible outside a specific platform – how much can we guarantee this protection?
- Learning. How fast can we learn enough about this framework to use it? Alternatively: how fast can we get people who want to help up to speed in it?
- Speed. How fast can we build something in this framework (we have 6 months to build a pilot system).
- Libraries. How many useful libraries are available (so we don't have to spend time building unnecessary stuff).
- Tools. How good are its instructions, IDEs, debuggers, test tools, database access?
- Adapters. How easy is it to run and connect analysis tools and to APIs from the framework? Most of the tools found so far are written in Python, and occasionally in R.
- Bandwidth. We need to be able to use this in a low-bandwidth environment.

- Pretty. We've been specifically asked to make the platform pretty. I'm sure there's a metric for that out there somewhere.

Feature	Rails	Django	Lift	Struts	Drupal	ASP
Help	Active community	Active community				
People	Existing UN volunteer teams				Sought after and difficult to obtain	
Privacy	Plug-in security					
Learning						
Speed	Fast				Takes time	
Libraries						
Tools	Multiple test tools	Unittest. Test client.		Unit tests	SimpleTest	Unit tests
Adapters	Has wrappers for Python	Natural for Python tools			Callout to python scripts. And Drupy!	
Bandwidth						
Pretty						

Well that's the matrix for later, when we have a longer development cycle and resources. For now, given the speed and people constraints, it comes down to a choice of two: Rails or Django. Each wins on different scores: Django on its python background (and hence really simple adapters), Ruby on its toolset, prettiness and available people. Which is what it comes down to in the end. We've been asked for something pretty and fast, and there isn't the luxury now to choose Scala's simplicity and security, or Drupal's community tools and Python adapters.   Perhaps when the open data Drupal project is up and running, we can share what we've learnt about needs and tools from this project with them, and start to build something *really* special for the world.

References:

http://www.embracingchaos.com/2010/05/choosing-a-web-framework-python-django-vs-ruby-on-rails.html
http://developmentseed.org/blog/2007/nov/26/drupal-meet-python
http://drupy.net/
http://www.assembla.com/wiki/show/liftweb