Uploading some specific features but not all developed features from development to live server

Asked 13 years, 9 months ago Modified 13 years, 7 months ago

Viewed 184 times Part of PHP Collective

6

This is more or less related to project management and also with every developer. How you guys handle this situation when you have developed many features on development site and all are tested by client and ready to go live.

These features have some code in common files ie. One PHP file have the code for one feature as well as one other feature.

But client will ask you to upload only 2 feature out of 10 or 15. Files are common if you upload that file directly will leads to error problems because they have code for other features. If you upload all updated files then all feature will be live.

A possible way is go back and comment out that feature which is not needed live for now from common files. But there is possiblities to forgot to comment anywhere else.

This is also not a good way and at last client will say what happen everything was tested on development server

and why these bugs and errors are introduced on live server. This will reduce the faith on developers.

I faced this problem many times and could not found any good way to avoid these issues. So I am thinking that you guys also facing or faced this problem.

I am thinking versioning system can help here.

How you guys are handling this?

Could you share ideas?

PHP php deployment project-management

Share Follow

edited Mar 22, 2011 at 15:20

asked Mar 22, 2011 at 5:26

Shakti Singh

86.3k ● 21 ● 139 ● 155

3 Answers

Sorted by: Highest score (default)

10

+50

The situation you are describing is impossible to manage sanely. I don't believe it would be possible to make this situation work, but the real question is why would you want to?

There are a number of issues with the scenario you describe, but the core issue is really this. You are testing one thing, and deploying another. You acknowledge in your question the interconnected nature of changes. In reality it is even more difficult than you describe. You simply cannot know how a system will behave when you try and deploy parts of a tested solution. Why test it at all?

The only sensible solution I can see is to have a sandbox environment where new features are demonstrated. However keep your test enviornment only for testing stuff that will go live. So in your example the one or two features are in test, ready to be signed off for prod, and the other featues are locked in the sandbox.

This leads to the next problem, which is managing your source code. I don't see any sane strategy for managing the arbirtrary inclusion of features from a code base. Even under the mostflexible system I know, Perforce, any branching straegy would require awful resolves on merges as you try to move stuff in and out.

I have seen this happen, and believe me it gets very ugly.

I suggest you come up with a better solution. Talk to your client and change the way things are done. It will be better for you, and in the long run better for them.

Share Follow

answered Mar 22, 2011 at 7:32

Phil Wallach

3,318 ● 21 ● 19

1 +1 for succinctly identifying the core issue: "You are testing one thing, and deploying another." – Mike Sherrill 'Cat Recall' May 30, 2011 at 13:35

2 +1 Bad planning from the company or bad request from the client (probably the latter ;-)) – Capsule May 30, 2011 at 16:05

3

A solution could be to use cheap version branching as provided by VCS such as Git or Mercurial. The project would consist in many feature branches used to develop said features and build branches where feature branches would be merged and adhoc fixing would take place. When a build branch is ready for test, it is tested, fixed if needed and then the build branch is shipped to production platform. When features have been validated, the build branch can be merged into remaining feature branches so the branches under development can integrate the "official" changes. To sum up, the application is custom built from existing feature branches as needed.

Share Follow

answered Mar 22, 2011 at 15:38

alfmartinez
199 • 3

2

One reasonably sane way to manage this on the code level is to isolate each feature into a plugin. Then you can add/remove features on-demand by simply enabling or disabling corresponding plugins.

But this solution has certain costs:

1. Time to develop and test plugin engine for your app
2. You need to test every plugin configuration (set of enabled plugins and their versions) that is going to be deployed. Otherwise there's a risk that this specific set is not compatible and end users would be first to see resulting crash, or data loss, or some other horror
3. Additional time to wirte plugins the way that they're minimally dependent on each other.

It's usually worth it only if you have many clients with different needs. In your case, I'd recommend explaining cost of separately enabling features to your client to see if they really need it this hard. Most likely, they don't

Share Follow

answered May 24, 2011 at 11:24

Alexander Lebedev
6,044 • 1 • 21 • 30