Continuous Testing Server

The server is at [http://quality.tiki.org](http://quality.tiki.org)

Automated tests on trunk:

Background

Because [Tiki Selenium tests](http://quality.tiki.org) take so long to run, it is not practical for anyone to run them all before committing.

So, we need some sort of program that will continuously test the newest version of Tiki, and report on failed tests. Basically, it would do something like this:

While true:

- Sleep for 5 minutes
- Do a SVN update to see if there have been changes since the last time the tests were run.
  - If no changes, GO BACK TO START OF LOOP
- Run all tests
- If all tests are OK, GO BACK TO START OF LOOP
- If failing tests were already failing in last iteration fo the loop, GO BACK TO START OF LOOP
- Otherwise, send an email notification to a list of folks (ex: Quality team), with details about:
  - Which tests started failing in this iteration
  - Which commits were updated in this iteration

Of course, it would be relatively easy to implement this in Php inside Tiki. But I was wondering if it would make more sense to use an existing continuous framework like Cruise Control.

Does anyone here have experience with Cruise Control or any other continuous integration and testing frameworks?

Roadmap

Here's a list of things to do if we are to make progress towards that.

- Make sure all tests can be run smoothly on any machine
  - As of 2009-10-15, there does not seem to have a AllTests or phpunit XML file that collates all tests (both unit and selenium ones) into a single suite that could be executed by the continuous testing server.
  - Also, the Selenium tests may have some dependancies on pathes that only work on the machines of Alain Désilets and Marta Stojanovic
    - For example, URL of the localhost to use for testing. This one was dealt with by AD in a
recent commit. But there may be others lurking.

- Establish the data flow that the testing server should support. We outlined a possible one in the Background section, but it may not be correct.
- Investigate php under control and cruise control, to see if they could be used to run the tests as specified in the above data flow.
  - If not, see if a simple php script could be written to do this.
- Procure and configure a machine to run the tests continually, and start using it.

Continuous Integration
How to improve the release process
http://www.linuxjournal.com/content/sourceclears-commit-watcher
https://goreplay.org/

Related

- Continuous Integration
- TikiTests