QA - action #49502

Automatic validation test on github PRs

2019-03-20 10:50 - cachen

Status:	New	**Start date:**	2019-03-20
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	future		

Description

Purpose:
-To reduce the unexpected disturbance by other's code change especially on those sharing lib/module
-To make QA developers and code reviewers life much easier

What we have:
-QSF team had an implementation which can manually trigger the expected validation test on OSD, description is in http://open.qa/docs/#_triggering_tests_based_on_an_any_remote_git_refspec_or_open_github_pull_request

What are missing:
-ideally to make it fully automated, it need a review bot integrate to github can automatic monitor the PR status -> trigger/schedule validation tests on OSD or Staging area -> update status to github

The points from coolo:
-easiest to do is in python
-https://developer.github.com/v3/repos/statuses/ is the github api to set a status report and there are python apis for it
-https://github.com/openSUSE/obs-tools/tree/master/pull_request_package is a ruby bot that obs team maintains to set a status in their PRs if the package builds
-so you would poll the pull requests if the status is set - and if it isn't, schedule a set of openQA tests the way okurz mentioned, wait for them to finish and update the github status

What we need:
-contributor who is interested on python and github bot for automation :)

Related issues:

Related to openQA Project - coordination #58184: [saga][epic][use case] full ...	**Blocked**	**2018-03-23**
Related to openQA Project - action #52004: Enable SUSE partners to test SLE u...	**Resolved**	**2019-05-26**

History

#1 - 2019-03-20 11:01 - cachen

Add Rodion and Matthias(stand-in Oliver as PO) in the loop.

@all, feel free give suitable flag to category

#2 - 2019-03-20 11:14 - riafarov

We already make some steps in #37958. In short: for the next sprint we are going to move staging tests to the new scheduling mechanism so no unintended changes to the schedule affect staging. The next step would be to use that list of the module to provide some warning in case PR modifies any of test modules executed in staging, so we can request VR for it.

Idea which we have in parallel is some kind of intermediate state for the master branch, for which we will run actual openQA job, and only if that succeeds - move it to the production.

Does it cover motivation of this ticket?

#3 - 2019-03-21 06:52 - cachen

riafarov wrote:

> We already make some steps in #37958. In short: for the next sprint we are going to move staging tests to the new scheduling mechanism so no unintended changes to the schedule affect staging. The next step would be to use that list of the module to provide some warning in case PR modifies any of test modules executed in staging, so we can request VR for it.
>
> Idea which we have in parallel is some kind of intermediate state for the master branch, for which we will run actual openQA job, and only if that succeeds - move it to the production.

> Does it cover motivation of this ticket?

IMO #37958 is part of this ticket motivation as well as #47441 "[functional][u][timebox:8h] test code feedback - integrate bots to test distribution on github",

After read the original ideas in #44327 and several discussion in emails with coolo and Oliver, I distinguish the self-test(or validation) into 2 stages as the fully-automation requirements, but feel free correct me:

1)Before commit:
-locally pre-checking(testing)to help QA developer for maximum quality of the modified codes before commit, tuning their codes with some standard(such as perl-tidy), then they don't need fix/rebase time to time after the committing,
-we also can put in this stage the "automatic analyze the modified codes for which modules in openQA tests, and automatic generate the according tests, and help QA developer automatic trigger(such as "openqa-clone-custom-git-refspec $GITHUB_PR_URL $OPENQA_TEST_URL"), or just suggest the according tests should be validated before commit,
-roughly idea is this can be made by github hooks to such as pre-commit, and the test status can be automatic generated to such as prepare-commit-msg. For this, reviewers will feel easier.

2)On pull request: we still need bots and integrate to github,
-to analyze the modified codes for the affected modules, automatic assign/notify reviewers(the modified code's owner/maintainer for example)
-to read the test status, compare and schedule according test modules, trigger the additional tests, waiting finish and update status to PR

It looks to me, #37958#47441 are much more in the above stage 2) and can be extended? Currently I don't have much idea how the 2) be made, coolo has pointed to me 2 links in my description and "the hard part is actually deciding what to schedule". Here need you all's intelligence.

#4 - 2019-03-21 08:41 - cachen

therefore, before any implementation it need an overall consideration to a integrated design.

Add Sunny and Yifan as well, in case guys from them are also interested on this interesting automation project.

#5 - 2019-03-21 09:09 - riafarov

cachen wrote:

> riafarov wrote:
>
>> We already make some steps in #37958. In short: for the next sprint we are going to move staging tests to the new scheduling mechanism so no unintended changes to the schedule affect staging. The next step would be to use that list of the module to provide some warning in case PR modifies any of test modules executed in staging, so we can request VR for it.
>>
>> Idea which we have in parallel is some kind of intermediate state for the master branch, for which we will run actual openQA job, and only if that succeeds - move it to the production.
>>
>> Does it cover motivation of this ticket?
>
> IMO #37958 is part of this ticket motivation as well as #47441 "[functional][u][timebox:8h] test code feedback - integrate bots to test distribution on github",
>
> After read the original ideas in #44327 and several discussion in emails with coolo and Oliver, I distinguish the self-test(or validation) into 2 stages as the fully-automation requirements, but feel free correct me:
>
> 1)Before commit:
> -locally pre-checking(testing)to help QA developer for maximum quality of the modified codes before commit, tuning their codes with some standard(such as perl-tidy), then they don't need fix/rebase time to time after the committing,
> -we also can put in this stage the "automatic analyze the modified codes for which modules in openQA tests, and automatic generate the according tests, and help QA developer automatic trigger(such as "openqa-clone-custom-git-refspec $GITHUB_PR_URL $OPENQA_TEST_URL"), or just suggest the according tests should be validated before commit,
> -roughly idea is this can be made by github hooks to such as pre-commit, and the test status can be automatic generated to such as prepare-commit-msg. For this, reviewers will feel easier.
>
> 2)On pull request: we still need bots and integrate to github,
> -to analyze the modified codes for the affected modules, automatic assign/notify reviewers(the modified code's owner/maintainer for example)
> -to read the test status, compare and schedule according test modules, trigger the additional tests, waiting finish and update status to PR
>
> It looks to me, #37958#47441 are much more in the above stage 2) and can be extended? Currently I don't have much idea how the 2) be made, coolo has pointed to me 2 links in my description and "the hard part is actually deciding what to schedule". Here need you all's intelligence.
> I would say checking everything before even committing is an overkill, so yes, it's more focused on 2), but all CI checks we do with travis can be triggered manually in local setup. So it doesn't matter if we have working solution, it will be possible to trigger locally (will require CI setup similar to travis though). But yeah, it's good to know that we share same concerns, so can address it ;)

#6 - 2019-10-15 10:20 - okurz

- Related to coordination #58184: [saga][epic][use case] full version control awareness within openQA, e.g. user forks and branches, fully versioned test schedules and configuration settings added

#8 - 2020-11-12 15:59 - okurz

One note: If I understand different ideas right then you would appreciate automatic openQA verification runs for every PR in github.com/os-autoinst/os-autoinst-distri-opensuse/ . But I only see tests possible that run on a public instance for that public repo. That means we can not run SLE tests on openqa.suse.de as access is not possible for all contributors. I would not enable automatic SLE tests which are not available for all. Maybe we can even run all SLE Maintenance tests in the open just want to keep "pre-public-Beta" builds of new versions in development internal. I know that technically this is not a problem at all. This is really about politics, marketing and a proper process. We could ask responsibles within SUSE how to make it public and after the "political approval" we can do that. #52004 is the corresponding ticket for that.

#9 - 2020-11-12 15:59 - okurz

- Related to action #52004: Enable SUSE partners to test SLE using openQA in the open easily added

#10 - 2020-11-13 03:54 - llzhao

How about the code changes on projects "openQA / os-autoinst / os-autoinst-needles-sle"? Will this poo also cover the "Automatic validation" on their code/needle changes?

#11 - 2020-11-13 08:58 - okurz

llzhao wrote:

> How about the code changes on projects "openQA / os-autoinst / os-autoinst-needles-sle"? Will this poo also cover the "Automatic validation" on their code/needle changes?

Well, first, this ticket can be seen as more generic than just openQA as it is in the general "QA" context. Also there is no one currently even planning to implement this change as described here so it is undefined if the mentioned projects would be covered or not.

However I can state what I find feasible and useful. os-autoinst and openQA already have a good test coverage within their projects so I consider it "done" already. What can be improved a lot is test coverage for "exotic backends", e.g. the backends that are used for s390x, powerVM, bare-metal. This is something I am expecting from the power users that rely on these backends as this is Out-of-Scope for SUSE QE Tools.

For os-autoinst-needles-sle there are already some tiny static CI checks but not more. As long as needle changes are automatically submitted to the master branch by the use of the webUI of openQA I see no use in extending the existing CI checks as the minority of users that create merge requests would be punished for more problems that others have introduced working around the CI. The only feasible option I see to improve the validation of SLE needle changes would be to have all automatic commits from the webUI to go into a different branch, then create automatic merge requests with more checks and tests and review before accepting into the master branch or another "stable" branch which would be used for some or all products on osd.

#12 - 2021-04-29 09:19 - okurz

- Target version set to future