

Working in Teams

Chung-Kil Hur

(Credit: Byung-Gon Chun & Many Slides from UCB CS169
taught by Armando Fox and David Patterson)

SWPP, CSE, SNU

Outline

- Design Reviews and Code Reviews
- The Five R' s of Bug Fixing
- Pitfalls

Agile Design and Code Reviews??

Good Meetings: SAMOSAS



(Photo by K.S. Poddar. Used by permission under CC-BY-SA-2.0.)

- **S**tart and stop meeting promptly
- **A**genda created in advance; no agenda, no meeting
- **M**inutes recorded so everyone can recall results
- **O**ne speaker at a time; no interrupting talker
- **S**end material in advance, since reading is faster
- **A**ction items at end of meeting, so know what each should do as a result of the meeting
- **S**et the date and time of the next meeting

Minutes and action items record results of meeting, start next meeting by reviewing action items

Design/Code Reviews

- **Design review**: meeting where authors present design with goal of quality by benefiting from the experience of the people attending the meeting
- **Code review**: held once the design has been implemented
- In the Agile/Scrum context, since design and implementation occur together, they might be held every few iterations

Design/Code Reviews

- Shalloway*: formal design and code reviews often too late in process to make big impact
- Recommends instead have earlier, smaller meetings: “**approach reviews**”.
 - A few senior developers assist team in coming up with an approach to solve the problem
 - Group brainstorms about different approaches
- If do a formal design review, suggests 1st hold a “**mini-design review**” to prepare

*Alan Shalloway, *Agile Design and Code Reviews*, 2002,
www.netobjectives.com/download/designreviews.pdf

Code Reviews

Can Check Comments too

- Challenge: keeping comments consistent with changes, refactoring



- Code review is one place to ensure comments make sense

Comment Example

```
# Add one to i.
```

```
i += 1
```

```
# Lock to protect against concurrent access.
```

```
mutex = SpinLock.new
```

```
# This method swaps the panels.
```

```
def swap_panels(panel_1, panel_2)
```

```
  # ...
```

```
end
```


Comment Example

```
# Loop through every array index, get the  
# third value of the list in the content to  
# determine if it has the symbol we are  
looking
```

```
# for. Set the result to the symbol if we  
# find it.
```

```
# Good Comment:
```

```
# Scan the array to see if the symbol exists
```

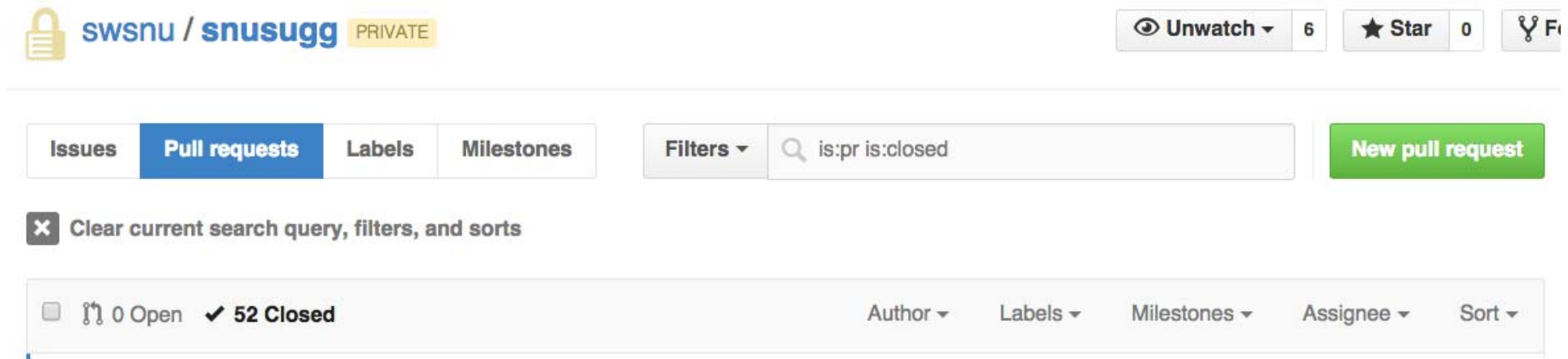
Advice on Comments

- Document **what is not obvious** from code
 - Don't just repeat what's obvious from the code
- **Raise level of abstraction**
 - Do think about what's not obvious at both the low level and the high level
- Explain **why** did something
 - Design issues that went through your mind while you were writing the code

Reviews and Agile?

- Pivotal Labs – pair programming makes review superfluous
- GitHub – *pull requests* replace code reviews
 - 1 developer requests her code to be integrated with code base
 - Rest of team see request and determines how affect their code
 - Any concern leads to online discussion
 - Many **pull requests**/day
=> many **minireviews**/day

Code Review Example (Good)



The screenshot shows the GitHub interface for a pull request in the repository `swsnu / snusugg`, which is marked as `PRIVATE`. The top navigation bar includes links for `Unwatch` (6), `Star` (0), and a fork icon. Below this, the `Pull requests` tab is selected, with other tabs for `Issues`, `Labels`, and `Milestones`. A search bar contains the query `is:pr is:closed`, and a green `New pull request` button is visible. A link to `Clear current search query, filters, and sorts` is provided. The main content area shows a summary of pull requests: `0 Open` and `52 Closed`. At the bottom, there are dropdown menus for `Author`, `Labels`, `Milestones`, `Assignee`, and `Sort`.

<https://github.com/swsnu/snusugg/pull/49>

Code Review Example (Bad)



spica commented on Dec 12, 2014



마차노트 이렇게 테스트할거만이야

Labels



bug

Error #12. Name, price which is too long overlaps its text form (Web) #123

Edit

New issue

 Closed

spica opened this issue on Dec 12, 2014 · 0 comments

Which expression statement regarding Design Reviews and Meetings is FALSE?

- ☐ Intended to improve the quality of the software product using the wisdom of the attendees
- ☐ They result in technical information exchange and can be highly educational for junior people
- ☐ Design reviews can be beneficial to both presenters *and* attendees
- ☐ Serving food like Samosas is vital to success of a good meeting

Fixing Bugs: The Five R's

No Bug Fix Without a Test!

- **Report**
 - **Reproduce and/or Reclassify**
 - **Regression test**
 - **Repair**
 - **Release the fix (commit and/or deploy)**
-
- Even in non-agile organizations
 - But, existing agile processes can be *adapted* to bug fixes

Report

- GitHub “issues” feature
- Full-featured bug tracking, eg Bugzilla, JIRA
- *Use the simplest tool that works for your team & project scope*

Reclassify? or Reproduce + Repair?

- Reclassify as “not a bug” or “won’t be fixed”
 - Reproduce with *simplest possible* test, and add it to regression
 - minimize preconditions
- No bug can be closed without a test.*
- Release: may mean either “pushed” or “deployed”

Which type of bug could you *not* create a regression test for?

- Bug that depends on behavior of external service
- Bug that depends on time of day, day of year, etc.
- Bug that is statistical in nature (“Player must win jackpot at least X% of the time”)
- You can reproduce any of these dependencies in regression tests

Pitfall

- Pitfall: Dividing work based on the software stack rather than on features
- E.g., Front-end specialist, back-end specialist, customer-liaison, ...
- Better for each team member to deliver all aspects of a chosen story
- Better for app, better for career growth

Open Source Project Case Study: Apache REEF

REEF

- Big Data Operating System Core
- Started in late 2012, Apache Incubating in August 2014
- 22 PMC members
 - SNU, Microsoft, SK Telecom, UCLA, Pure Storage, UC Berkeley, University of Washington

References

<http://reef.incubator.apache.org/>

<https://cwiki.apache.org/confluence/display/REEF/Home>

Apache Way

- Incubating => Top-level
- “Meritocracy”
- Project organization
 - Project Management Committee (PMC)
 - Committer
 - Contributor

Issue (Feature/Bug) Handling

Class Project

- Github issue registration
- Discussion (design / impl)
- Coding
- Github pull request
- Github code review
- Code Merge
- Close the issue


Apache REEF

- [JIRA](#) issue registration
- Discussion (design / impl) through [JIRA](#)
- Coding
- Github pull request
- Github code review
- [Apache](#) code merge
- Close the [JIRA](#) issue

Design Discussion Example

- Tang Namespace (Gyewon Lee)


<https://issues.apache.org/jira/browse/REEF-31>

▼  Gyewon Lee added a comment - 10/Nov/14 17:29

In 6, I said

"I think `getBinding()` could take a namespace's name for its argument" but that's not what I intended. What I suggest is `getBinding()` contains only name for the target namespace so it does not have direct reference to the `ConfigurationBuilder`.

Thanks
Gyewon
[Reply](#)

▼  Markus Weimer added a comment - 10/Nov/14 18:17

Regarding #6: How about adding a method like this to `ConfigurationBuilder`:

```
public void shareBinding(Name<?> whatIsBeingShared, String...  
    namespaces);  
public void shareBinding(Class<?> whatIsBeingShared, String...  
    namespaces);
```

Add utility function in Tang to parse command lines

This would allow users to tell Tang to share the given instance with the given namespaces. If more than one of the given namespaces already binds this name or interface, the call throws a `BindException`. Another idea would be to expose a source namespace:

```
public void shareBinding(Name<?> whatIsBeingShared, String
```

Code Pull Request Example (JIRA: REEF-30)

[REEF-30] Run REEF on Mesos #52

 Closed

johnyangk wants to merge 14 commits into `apache:master` from `johnyangk:REEF-30`

 Conversation 33

 Commits 14

 Files changed 35



johnyangk commented on Jan 19

JIRA: [REEF-30]: <https://issues.apache.org/jira/browse/REEF-30>

reef_on_mesos

What's in it

- reef-runtime-mesos package
- bin/runmesostests.sh (for running reef-tests on Mesos clusters)
- HelloREEFMesos in reef-examples
- MesosTestEnvironment in reef-tests
- pom.xml changes(adding reef-runtime-mesos) in root, reef-tests, reef-examples

How to test(reef-tests) it


Pretty similar to running reef-tests on YARN.

1. run mesos daemons (<http://mesos.apache.org/gettingstarted/>)
2. At \$REEF_HOME, "mvn clean install -DskipTests"
3. At \$REEF_HOME/reef-tests, "mvn jar:test-jar"
4. At \$REEF_HOME, "bin/runmesostests.sh \$YOUR_MESOS_MASTER_IP"

Code Review Example

(JIRA: REEF-30)

.../runtime/mesos/util/HDFSConfigurationConstructor.java [View full changes](#)



((11 lines not shown))

11

+ *

12

+ * Unless required by applicable law or agreed to in writing,

13

+ * software distributed under the License is distributed on an

14

+ * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY

15

+ * KIND, either express or implied. See the License for the

16

+ * specific language governing permissions and limitations

17

+ * under the License.

18

+ */

19

+package org.apache.reef.runtime.mesos.util;

20

+

21

+import org.apache.hadoop.conf.Configuration;

22

+import org.apache.reef.tang.ExternalConstructor;

23

+

24

+import javax.inject.Inject;

25

+

26

+public final class HDFSConfigurationConstructor implements ExternalConstructor<Cor



markusweimer added a note on Jan 20

We have a very similar class in the YARN runtime. It might be a good idea to move it to `reef-utils-hadoop` and reuse it here instead of this one.



johnnyangk added a note on Jan 20



Can I create a separate JIRA issue for this? It may be good to move this and the one in the YARN runtime together.

Code Release

(0.10.0-incubating led by Byung-Gon Chun)

From	Byung-Gon Chun <bgc...@gmail.com>
Subject	[RESULT] [VOTE] Release Apache REEF 0.10.0-incubating
Date	Thu, 15 Jan 2015 00:20:26 GMT

The release vote passes with 3 +1 binding votes from Justin, Alan, and Chris and no -1 vote.

We already addressed the comments about DISCLAIMER and NOTICE in our master branch for our next release.

Thanks to everyone involved in making this release happen!

On Thu, Jan 15, 2015 at 8:53 AM, Chris Douglas <cdouglas@apache.org> wrote:

```
> +1
>
> Checksum and signature match, RAT OK, source tarball builds cleanly.
> LICENSE/NOTICE lgtn, and I agree with Marvin w.r.t. the missing
> DISCLAIMER (can be fixed in subsequent releases). -C
>
> On Sat, Jan 10, 2015 at 7:14 PM, Byung-Gon Chun <bgchun@gmail.com> wrote:
> > The Apache REEF PPMC has voted to release Apache REEF 0.10.0-incubating
> > based on the release candidate described below. Now it is the IPMC's turn
> > to vote.
> >
> > Here's the PPMC voting result (five binding +1 votes):
> >
> > http://mail-archives.apache.org/mod_mbox/incubator-reef-dev/201501.mbox/%3CCADq0cj7YNbVLYX5
> >
> > The source tarball, including signatures, digests, etc can be found at:
> > http://people.apache.org/~bgchun/apache-reef-0.10.0-incubating-rc1/
> >
> > The Git tag is release-0.10.0-incubating-rc1
> > The Git commit ID is 76147ea7a4a7b6114bee9d1b8b4d1d588b4f0ce2
> >
> > https://git-wip-us.apache.org/repos/asf?p=incubator-reef.git;a=commit;h=76147ea7a4a7b6114be
> >
> > Checksums of apache-reef-0.10.0-incubating-rc1.tar.gz:
> > MD5: e112edbfe15a67710ad697ab529c4816
> > SHA1: 654dc4b7a386ed17e1a8084aa45682ac6c17f159
> > SHA512:
> >
> a92a9f6eb4c6c25a3a83843bcd39da462107a10d573705e977744afa498a25a8e0c508d97480661b5ddb5724edc
```

Summary

- Version Control : Git
- Design and Code Reviews
- Fixing Bugs: The Five Rs
- When project done, take time to think about what learned before leaping into next one
 - What went well, what didn' t, what do differently