## Using an extra tool for code review

- Bitbucket as an **example** on how to make code reviews more collaborative/interesting/efficient/useful
- It would lead to **better communications and more knowledge sharing** while delivering to the business

## Why code reviews matter

- Code reviews share knowledge
- Code reviews make for better estimates
- Code reviews enable time off
- Code reviews mentor newer engineers

Source: https://www.atlassian.com/agile/code-reviews

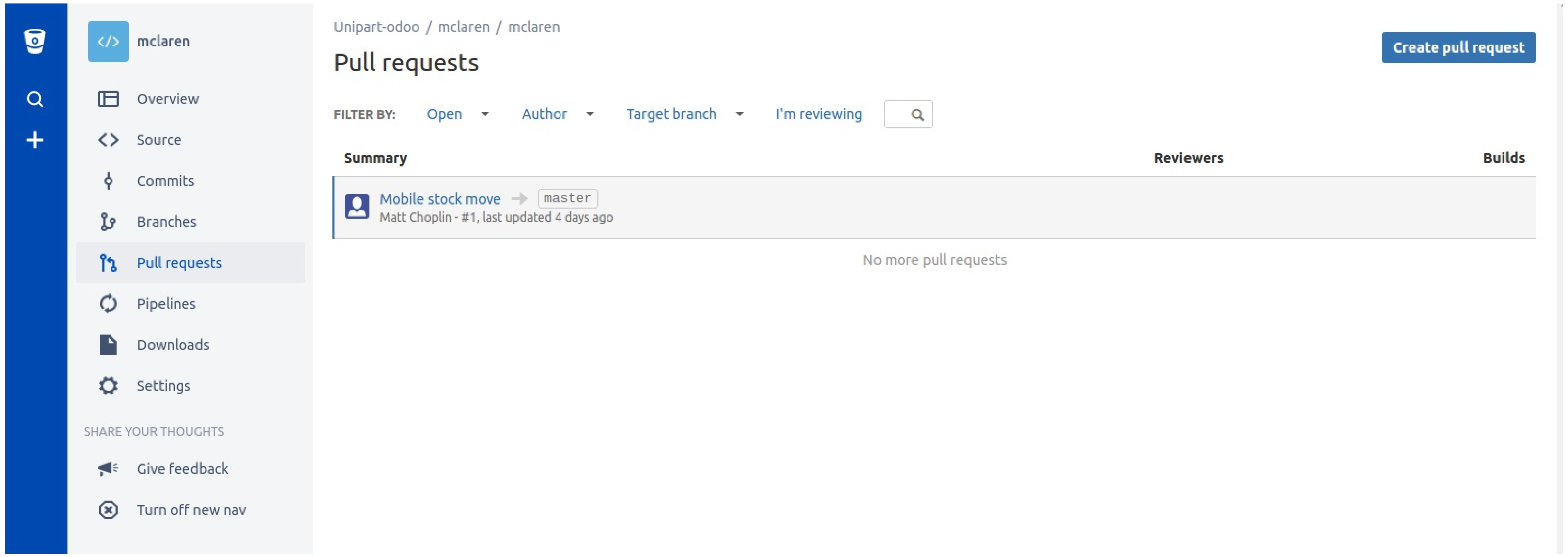
## But code reviews take time!

Use the following to optimise for that

- Share the load
- Review before merging
- Use peer pressure to your advantage

Home

## What it looks like (1)



Unipart-odoo / mclaren / mclaren

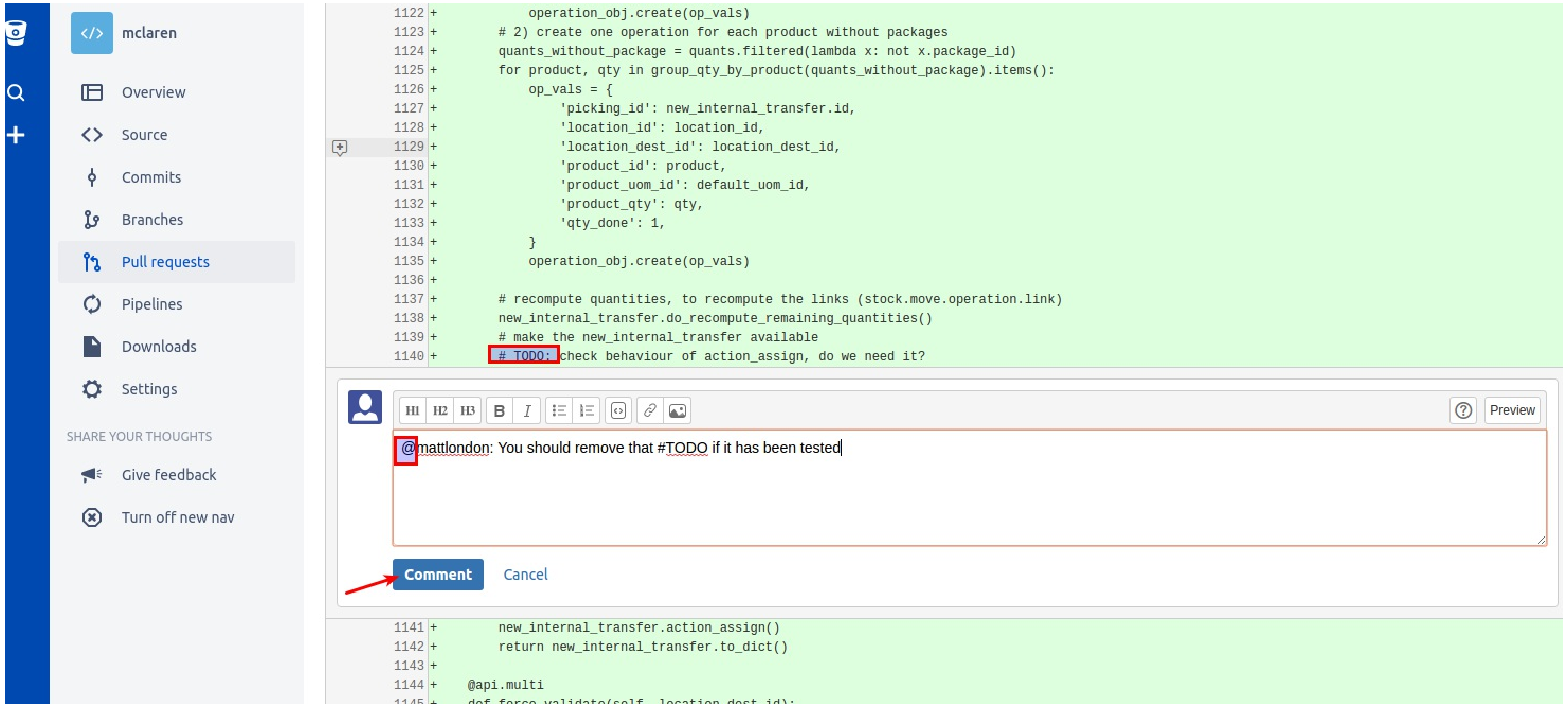
Pull requests

| Summary | Reviewers | Builds |
|---|---|---|
| Mobile stock move → master<br>Matt Choplin - #1, last updated 4 days ago | | |

No more pull requests

List of pending PR

# What it looks like (2)



```
+            operation_obj.create(op_vals)
+        # 2) create one operation for each product without packages
+        quants_without_package = quants.filtered(lambda x: not x.package_id)
+        for product, qty in group_qty_by_product(quants_without_package).items():
+            op_vals = {
+                'picking_id': new_internal_transfer.id,
+                'location_id': location_id,
+                'location_dest_id': location_dest_id,
+                'product_id': product,
+                'product_uom_id': default_uom_id,
+                'product_qty': qty,
+                'qty_done': 1,
+            }
+            operation_obj.create(op_vals)
+
+        # recompute quantities, to recompute the links (stock.move.operation.link)
+        new_internal_transfer.do_recompute_remaining_quantities()
+        # make the new_internal_transfer available
+        # TODO: check behaviour of action_assign, do we need it?
```

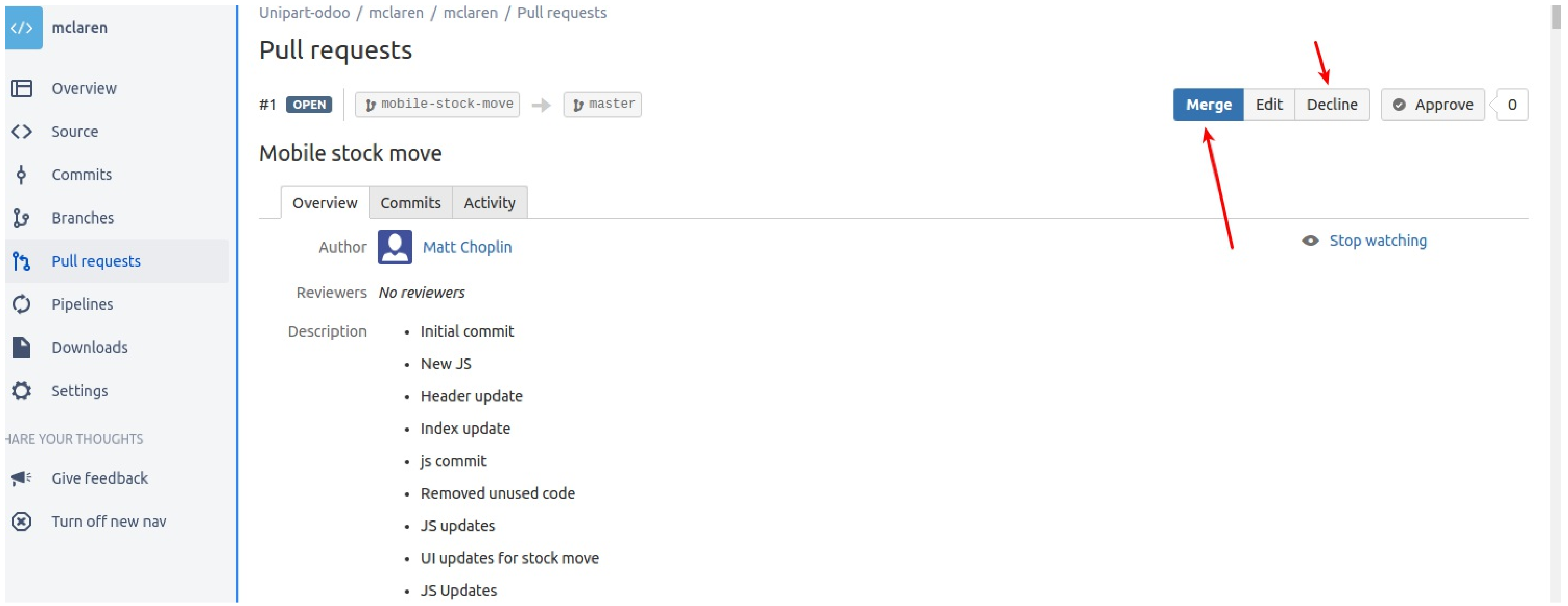
@mattlondon: You should remove that #TODO if it has been tested

```
+        new_internal_transfer.action_assign()
+        return new_internal_transfer.to_dict()
+
+    @api.multi
+    def force_validate(self, location_dest_id):
```

When we comment on a PR, the @mention will receive a notification

Home

# What it looks like (3)



Mobile stock move

Author Matt Choplin

Reviewers *No reviewers*

Description
- Initial commit
- New JS
- Header update
- Index update
- js commit
- Removed unused code
- JS updates
- UI updates for stock move
- JS Updates

Merging and processing PR

## Configuring Bitbucket from our existing repo...

- To make the repo point to a bitbucket repo:

```
git remote set-url origin https://github.com/USERNAME/OTHERREPOSITORY.git
```

- Create a team and link the repo to the team (free until 5 users): https://bitbucket.org/unipartodoo/mclaren (you need an invitation to access the code)

## How to make pull request

- Create a branch against master

```
git checkout -b my_feature_branch
```

- Commit
- Push to your feature branch
- Review: https://bitbucket.org/unipartodoo/mclaren/pull-requests/ (you need an invitation to access the code)

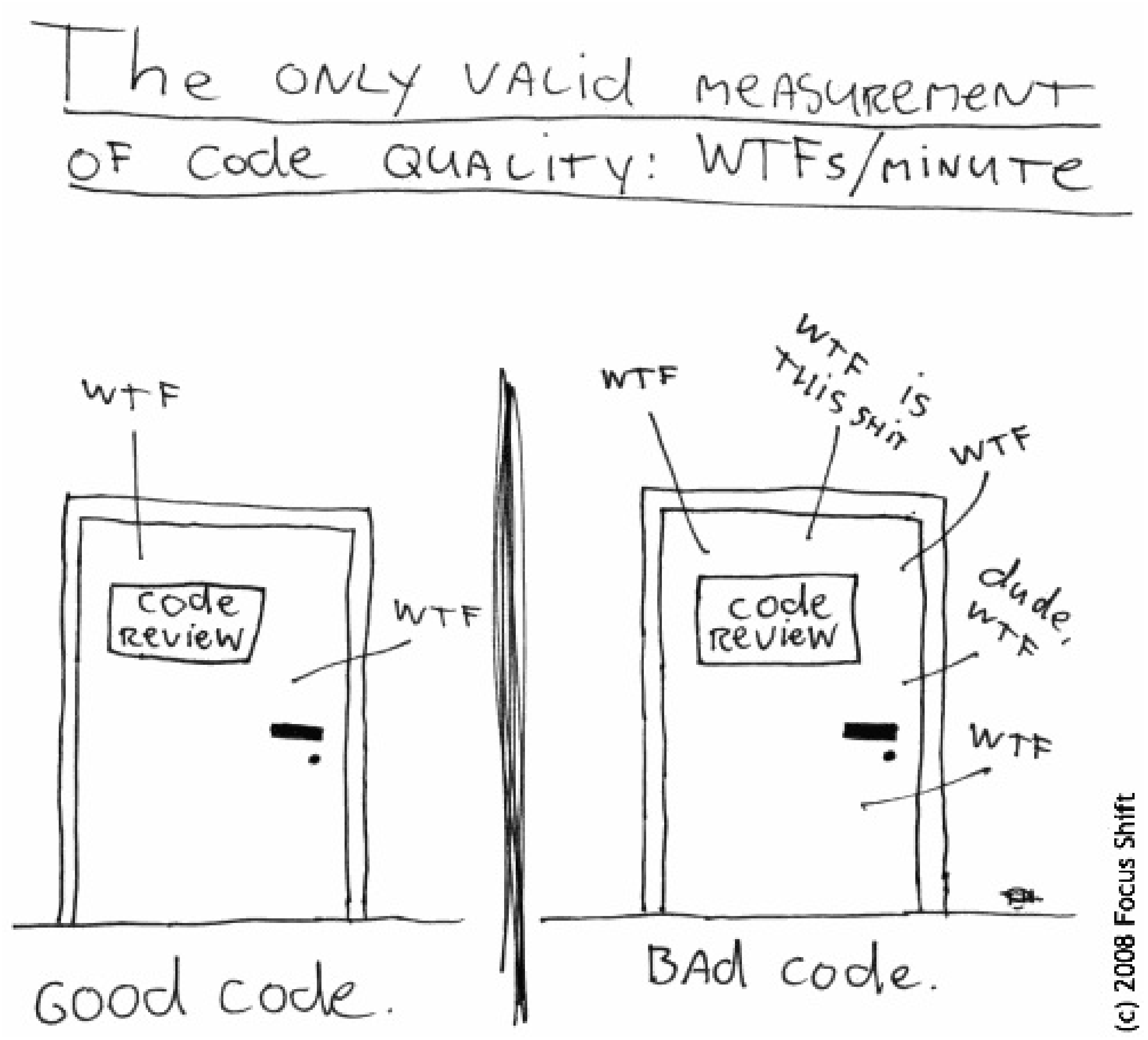
# The seven rules of a great Git commit message

1. Separate subject from body with a blank line
2. Limit the subject line to 50 characters
3. Capitalize the subject line
4. Do not end the subject line with a period
5. Use the imperative mood in the subject line
6. Wrap the body at 72 characters
7. Use the body to explain what and why vs. how

Source: https://chris.beams.io/posts/git-commit/

# How to make efficient code reviews

## In an image...



## Efficient code reviews (seriously)

- Ensure you have a clear definition of done
- Make action obvious: status and assignee
- Integrate your tools
- Use @mentions to keep conversations in one place

Source: https://www.atlassian.com/blog/software-teams/5-tips-great-code-reviews

## Next...

- branching strategy? Gitflow by Vincent Driessen
- CI using Bitbucket pipelines or others?