

Version control, issue tracking, and communication

Software Development, DIKU spring semester 2016

Presenter: Lars Yde, M.Sc., Engineering Manager, Milestone Systems

Contact

- ▶ Academic collaboration: thesis, B.Sc. project, dissertation, internship
 - ▶ Lars Yde, Engineering Manager : lay@milestone.dk (or larsyde@gmail.com)
 - ▶ John Madsen, Software Architect : jm@milestone.dk
- ▶ Milestone Systems, company website and information
 - ▶ www.milestonesys.com

Version control

Return to Zero



EEWeb.com

Version Control

- ▶ Purpose
 - ▶ Manage change in time (history) and space (branches)
- ▶ Benefits
 - ▶ Ease collaboration
 - ▶ Allow branching and continuous integration
 - ▶ Track ownership and change history
- ▶ Types
 - ▶ Local
 - ▶ SCCS
 - ▶ Centralized
 - ▶ E.g. Subversion
 - ▶ Distributed
 - ▶ E.g. Git

A Brief Timeline of Version Control Systems

1972	Source Code Control System (SCCS) is launched, developed in SNOBOL at Bell Labs by Marc Rochkind.
1982	Revision Control System (RCS) is released. RCS is still maintained by the GNU Project.
1990	Initial release of Concurrent Versions System (CVS) version control system
2000	Subversion (SVN) is launched by CollabNet.
2001	The first SVN source code is hosted.
2004	SVN version 1.0 is released.
26 March 2005	Bazaar (then "Baz") is released under Canonical's sponsorship.
7 April 2005	Git is released as an open source project headed by Linus Torvalds, the namesake of Linux.
19 April 2005	Mercurial project is launched a few days after Git, also designed for use with Linux development.
2008	The final stable release of CVS marks the end of an era (18 years).

Version control - definitions

- ▶ "Manage change"
 - ▶ Put files in a data store (file system, database, etc)
 - ▶ Optionally associate with metadata
 - ▶ Maintain deltas
 - ▶ Rinse & repeat
- ▶ Achieves
 - ▶ Traceability
 - ▶ Concurrency
 - ▶ Reversability

Terminology 101 [10]

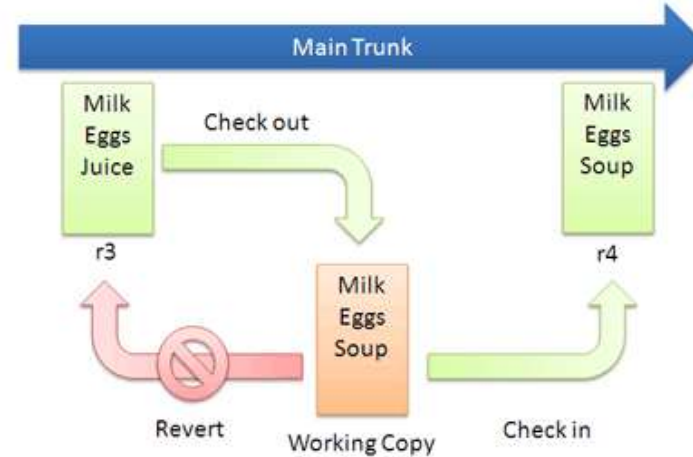
- ▶ Branch : A set of files under version control may be branched or forked at a point in time so that, from that time forward, two copies of those files may develop at different speeds or in different ways independently of each other.
- ▶ Change : A change (or diff, or delta) represents a specific modification to a document under version control.
- ▶ Checkout : To check out (or co) is to create a local working copy from the repository.
- ▶ Clone : Cloning means creating a repository containing the revisions from another repository.
- ▶ Commit : To commit (check in, ci or, more rarely, install, submit or record) is to write or merge the changes made in the working copy back to the repository.
- ▶ Conflict : A conflict occurs when different parties make changes to the same document, and the system is unable to reconcile the changes
- ▶ Head : Also sometimes called tip, this refers to the most recent commit, either to the trunk or to a branch
- ▶ Merge : A merge or integration is an operation in which two sets of changes are applied to a file or set of files
- ▶ Pull, push : Copy revisions from one repository into another
- ▶ Repository : The repository is where files' current and historical data are stored, often on a server
- ▶ Tag ; A tag or label refers to an important snapshot in time, consistent across many files
- ▶ Trunk : The unique line of development that is not a branch (sometimes also called Baseline, Mainline or Master)
- ▶ Working copy : The working copy is the local copy of files from a repository, at a specific time or revision

Version control - visual summary [12]

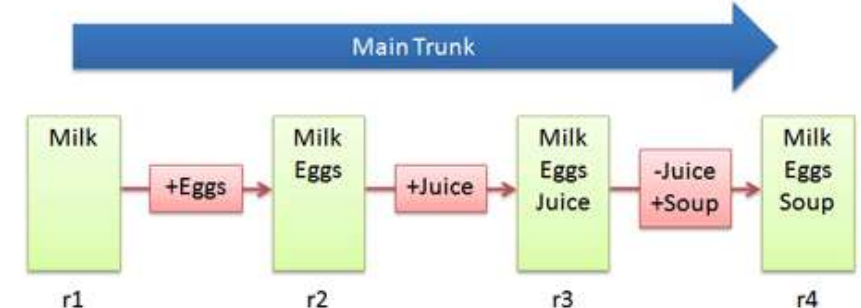
Basic Checkins



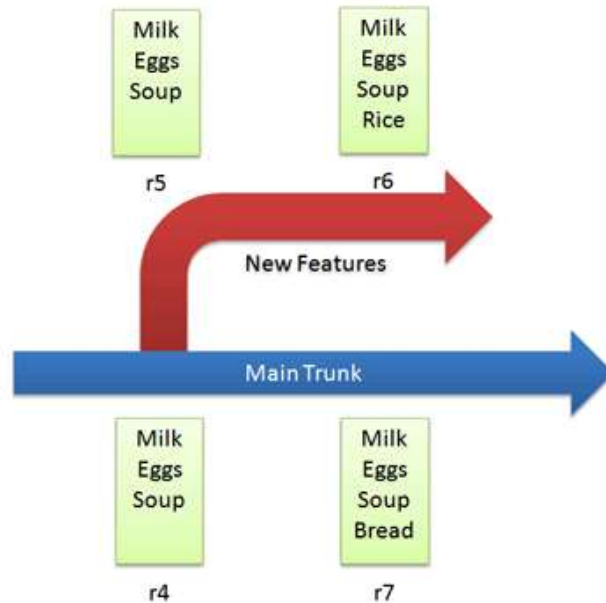
Checkout and Edit



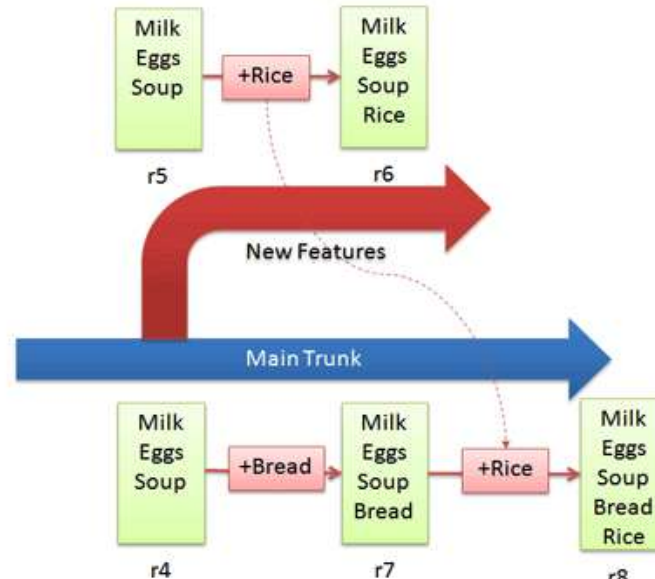
Basic Diffs



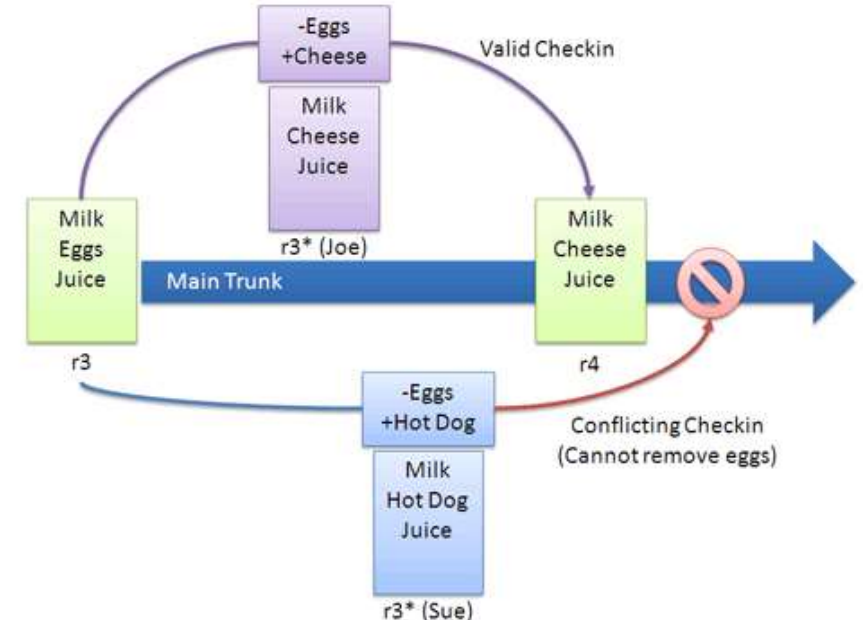
Branching



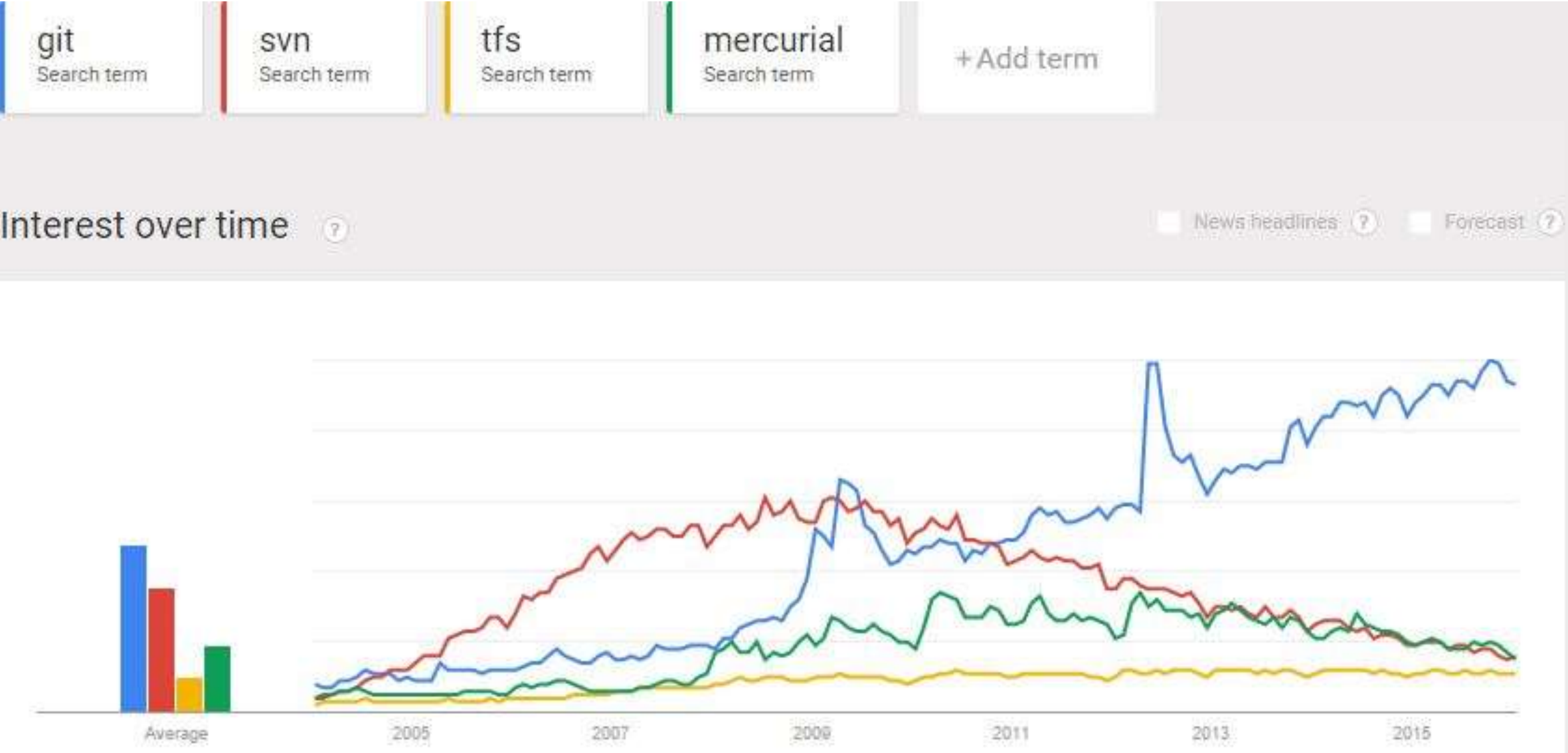
Merging



Conflicts



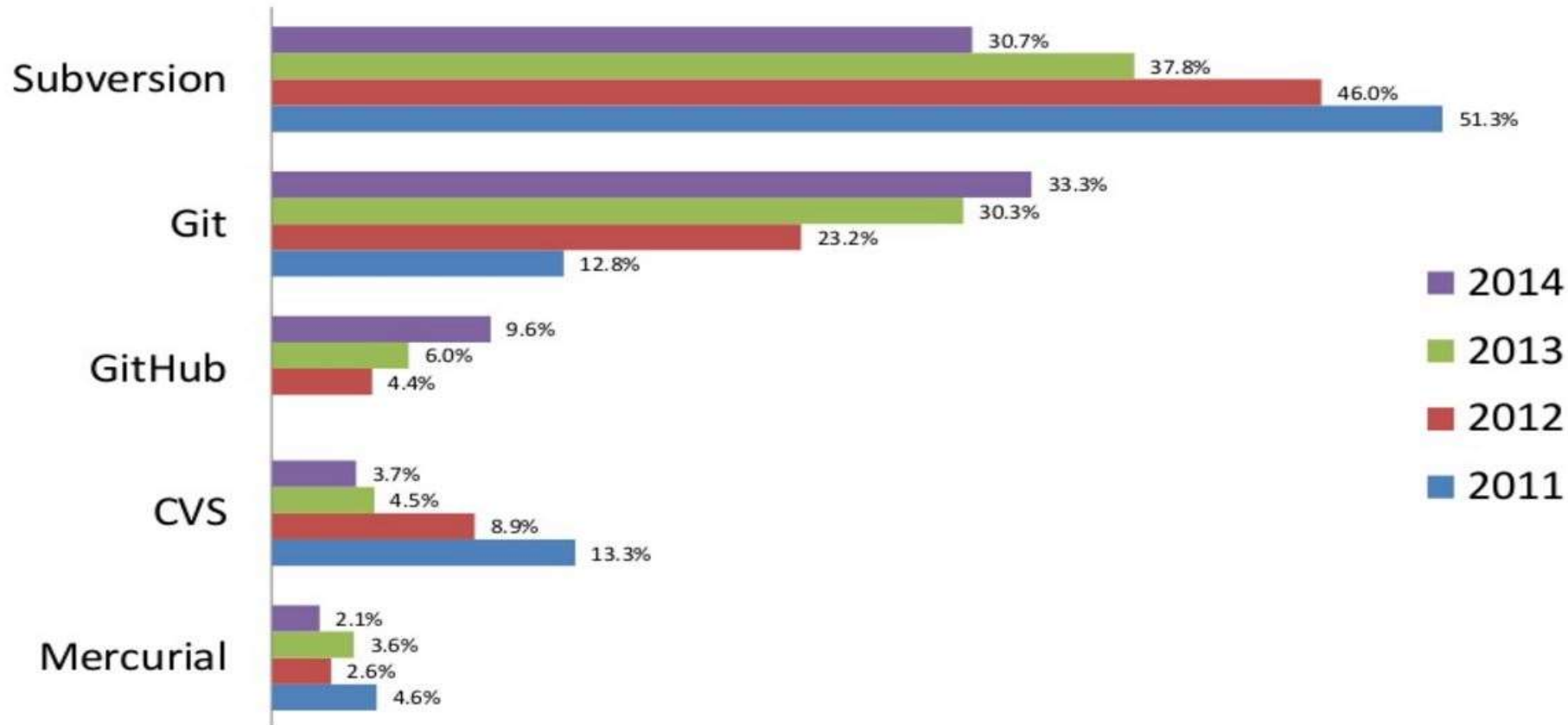
Version control system popularity - Google Trends



Primary Code Management



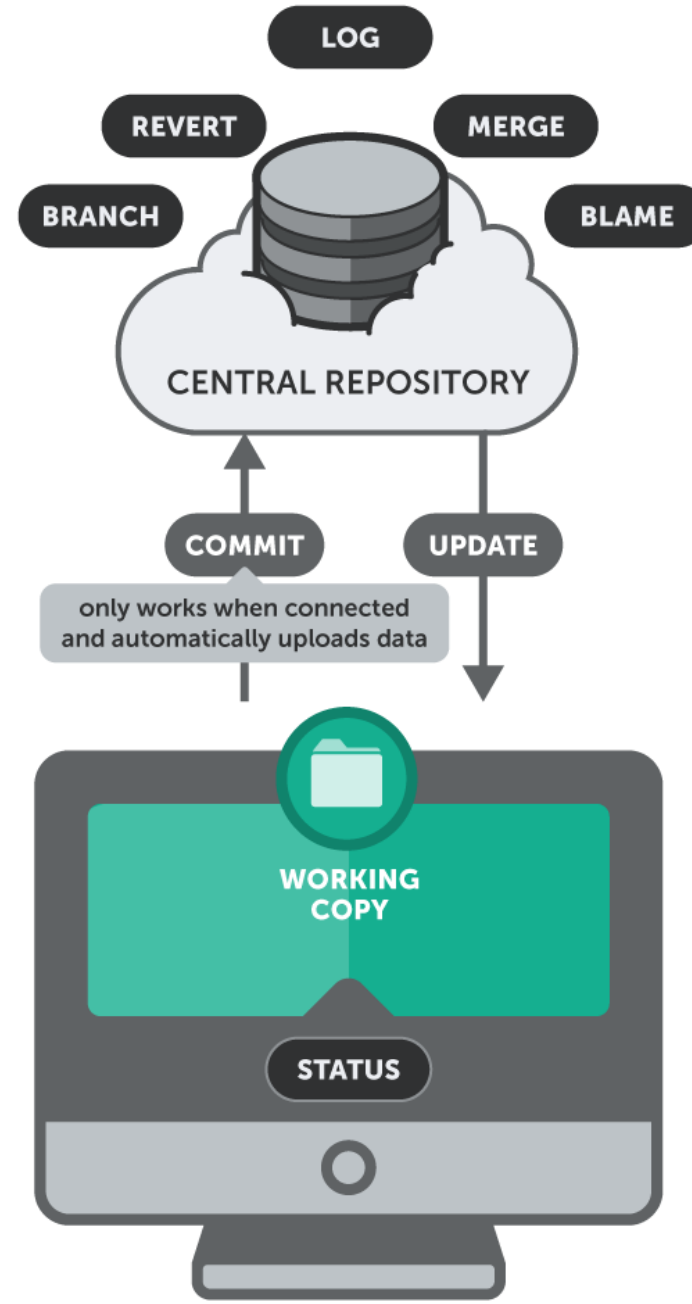
What is the primary source code management system you typically use? (Choose one.)



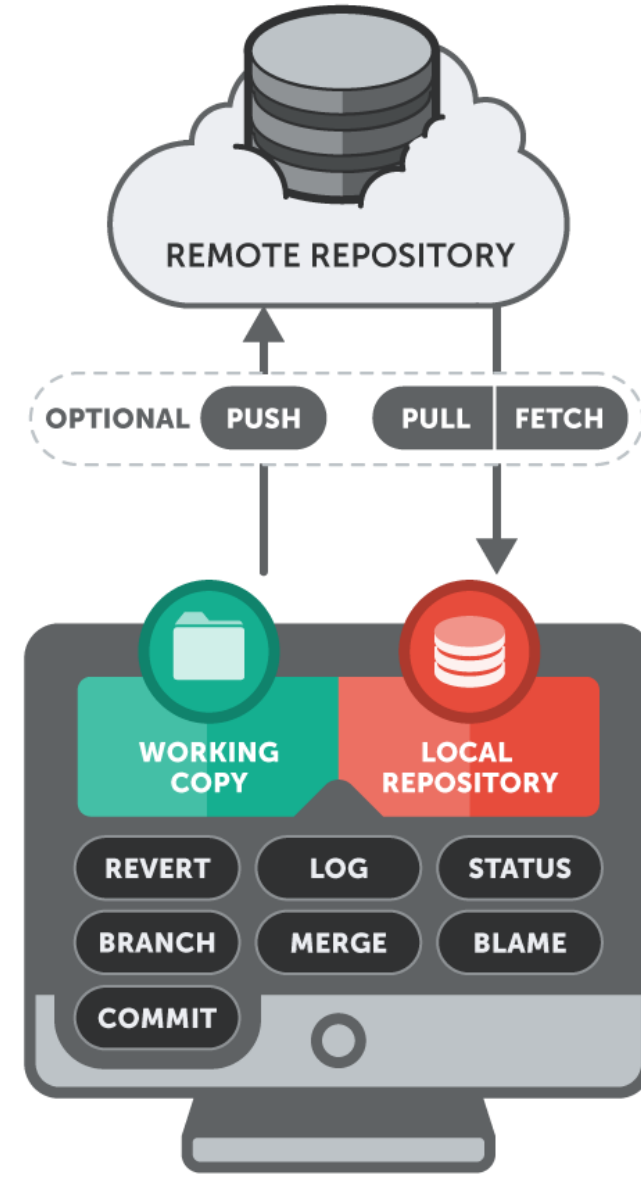
Why is git leading ?

- ▶ SVN is conceptually simpler and easier to learn, and has some good UI add-ons + it automatically backs up remotely
- ▶ But, git has some unique selling points
 - ▶ Faster
 - ▶ Works decentralized
 - ▶ Isolation (less FUD factor)
- ▶ Why is “the other DVCS” - Mercurial - lagging so far behind
 - ▶ Git’s a bit faster (written in C)
 - ▶ GitHub
 - ▶ Instant community because of Git origins in Linux world

SUBVERSION



GIT



Basic use scenarios

- ▶ "Save-as" version control
 - ▶ "MyCode.txt" => "MyCodeA.txt" => "MyCodeAA.txt" => ...you get the point
 - ▶ Use OS permissioning to "manage" concurrent access
 - ▶ Local version control systems (RCS) a refinement of this scheme [7]
- ▶ Central repository model (e.g. SVN) [8]
 - ▶ Basic cycle
 - ▶ SVN update => SVN add / delete / copy / move => SVN status / diff => SVN update / resolve => SVN commit => (goto start)
- ▶ Distributed repositories model (e.g. Git)
 - ▶ Git init => git add => git commit => git push => (git pull)

GitHub



- ▶ What is it ?
 - ▶ Git
 - ▶ Git with a web interface
 - ▶ Git hosting
 - ▶ Issue and request tracking system
 - ▶ Documentation system
 - ▶ A social media platform (feeds, followers, wikis, newsletter, social graph) ?

GitHub



- ▶ 12M users, 31M repositories
- ▶ Why ?
 - ▶ Free(mium)
 - ▶ Natural fit to open source / non-colocated collaboration
 - ▶ Popular! (network effects, winner takes all)
 - ▶ Web interface / accessibility
- ▶ Why not ?
 - ▶ Arguably a steep learning curve
 - ▶ Maybe overkill / bad fit for certain scenarios = don't make a sheeple decision

GitHub

- ▶ Tooling and interfaces
 - ▶ <https://git.wiki.kernel.org/index.php/InterfacesFrontendsAndTools>
- ▶ GUI clients (fat clients, various platforms)
 - ▶ <http://git-scm.com/downloads/guis>
 - ▶ Github desktop: <https://desktop.github.com/>
- ▶ Branching model
 - ▶ <http://nvie.com/posts/a-successful-git-branching-model/>



<demo>
<Subversion>

Available at
https://drive.google.com/file/d/0B3nCmF_pyhpQVC1ZYXFnTlVfa1k/view

(download for optimal viewing)

<demo>
<GitHub>

Available at
[https://drive.google.com/open?id=0B3nCmF_
pyhpQakN3dTYyNmhuZ1E](https://drive.google.com/open?id=0B3nCmF_pyhpQakN3dTYyNmhuZ1E)

(download for optimal viewing)

Demo still

https://github.com



[Pull requests](#) [Issues](#) [Gist](#)



Your repository "larsyde/my-new-test-repository" was successfully deleted.



Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Let's get started!



larsyde ▾



Welcome to GitHub! What's next? (2 days ago)

[Create a repository](#)

[Tell us about yourself](#)

[Browse interesting repositories](#)

[Follow @github on Twitter](#)

Your repositories 0

[+ New repository](#)

You don't have any repositories yet!
[Create your first repository](#) or [learn more about Git and GitHub.](#)

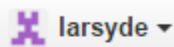
[Subscribe to your news feed](#)

Demo still

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner



Repository name

/ my new test repository



Great repository name

Your new repository will be created as **my-new-test-repository**-doodle.

Description (optional)

testing repository



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾


Add a license: **None** ▾



Create repository



Demo still

 [larsyde](#) / [my-new-test-repository](#)

Unwatch ▾ 1

★ Star 0

🍴 Fork 0

<> Code

! Issues 0

🔗 Pull requests 0

📖 Wiki

📶 Pulse

📊 Graphs

⚙ Settings

Filters ▾

🔍 is:issue is:open

Labels

Milestones


New issue

!

Welcome to Issues!

Issues are used to track todos, bugs, feature requests, and more. As issues are created, they'll appear here in a searchable and filterable list. To get started, you should [create an issue](#).

Demo still


 **larsyde / my-new-test-repository**


Unwatch 1


Star 0


<> Code


! Issues 0


 Pull requests 0

 Wiki

 Pulse

 Graphs

 Settings



get the todo list done

Write

Preview

AA ▾ B i “ <> 🔗 ☰ ☷ ✓ @ 🚩

please fix it

Attach files by dragging & dropping, [selecting them](#), or pasting from the clipboard.

M Styling with Markdown is supported

Submit new issue

Labels
None yet

Milestone
No milestone

Assignee
No one—assign yourself

Demo still

larsyde / my-new-test-repository

Unwatch 1

Star 0

Fork 0

Code

Issues 1

Pull requests 0

Wiki

Pulse

Graphs

Settings

get the todo list done #1

Edit

New issue

Open larsyde opened this issue just now · 0 comments



larsyde commented just now

Owner

please fix it

Labels

None yet

Milestone

No milestone

Assignee

No one—assign yourself

Notifications

Unsubscribe

You're receiving notifications because you authored the thread.

1 participant



Write

Preview

AA B i “ < > ⌨ ⋮ ⋮ ⋮ @

Leave a comment

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Close issue

Comment

Demo still

larsyde / my-new-test-repository

Unwatch 1

Star 0

Fork

<> Code

Issues 1

Pull requests 0

Wiki

Pulse

Graphs

Settings

testing repository — Edit

1 commit

1 branch

0 releases

1 contributor

Branch: master

New pull request

New file

Find file

HTTPS

https://github.com/larsyd

Download ZIP

Switch branches/tags

my test branch

Branches

Tags

Create branch: my test branch from 'master'


Latest commit b213be6 3 minutes ago

Initial commit 3 minutes ago

my-new-test-repository

testing repository

Demo still

 larsyde / my-new-test-repository

Unwatch ▾1

★ Star0

🍴 Fork

<> Code

! Issues 1

🔗 Pull requests 0

📖 Wiki

⚡ Pulse

📊 Graphs

⚙ Settings

testing repository — Edit

🕒 1 commit

🔗 1 branch

🏷 0 releases

1 contributor

Branch: master ▾

New pull request

New file

Find file

HTTPS ▾

https://github.com/larsyd

📄

📄

Download ZIP

Switch branches/tags

×

my test branch

BranchesTags

🔗 Create branch: my test branch
from 'master'


Latest commit b213be6 3 minutes ago

Initial commit 3 minutes ago

my-new-test-repository

testing repository

Demo still

 [larsyde](#) / [my-new-test-repository](#)

Unwatch ▾1

★ Star0

🍴 Fork0

<> Code

⚠ Issues1

🔗 Pull requests0

📖 Wiki


⚡ Pulse

📊 Graphs

⚙ Settings

Branch: [my-tes...](#) ▾ [my-new-test-repository](#) / [README.md](#)

Find fileCopy path

 **larsyde** Initial commit b213be6 4 minutes ago

1 contributor


3 lines (2 sloc) | 44 Bytes

Raw

Blame

History

🖨

 Edit this file

🗑

my-new-test-repository

testing repository

Demo still

larsyde / my-new-test-repository

<> Code

! Issues 1

🔗 Pull requests 0

📖 Wiki

📡 Pulsar

my-new-test-repository / README.md

<> Edit file

👁 Preview changes


1 ▾ # my todo list

2 1. todo number one

3 2. second todo

4 3. last todo

5



Commit changes

creating the todo list

getting it done

☒ Commit directly to the my-test-branch branch


☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)


Commit changes


Cancel

Demo still

<https://github.com/larsyde/my-new-test-repository/pulls>

 This repository Search

Pull requests Issues Gist + ▾ 

 larsyde / my-new-test-repository

Unwatch ▾ 1 Star 0 Fork 0

[Code](#) [Issues 1](#) [Pull requests 0](#) [Wiki](#) [Pulse](#) [Graphs](#) [Settings](#)


Filters ▾ is:pr is:open

Labels Milestones


New pull request

☐ [0 Open](#) [0 Closed](#)

Author ▾ Labels ▾ Milestones ▾ Assignee ▾ Sort ▾

 This repository

Pull requestsIssuesGist

 [larsyde](#) / [my-new-test-repository](#)

Unwatch 1

<> Code

! Issues 1

🔗 Pull requests 0

📖 Wiki


⚡ Pulse

📊 Graphs


⚙ Settings

Comparing changes


Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

 base: **master** ... compare: **my-test-branch**


✓ **Able to merge.** These branches can be automatically merged.


 **Create pull request** Discuss and review the changes in this comparison with others.



 **1** commit


 **1** file changed


 **0** commit comments

 **1** contributor

 Commits on Feb 04, 2016

  **larsyde** creating the todo list ...

 Showing **1 changed file** with **4 additions** and **2 deletions**.

6  README.md


<>📄V


...	...	@@ -1,2 +1,4 @@
1		-# my-new-test-repository
2		-testing repository
	1	## my todo list

Demo still

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you

 base: **master** ... compare: **my-test-branch** ✓ **Able to merge.** These branch



creating the todo list, fixes #1

Write

Preview

AA ▾ B i “ <> c

getting it done


Demo still

creating the todo list, fixes #1 #2

 **Open** larsyde wants to merge 1 commit into `master` from `my-test-branch`

 Conversation **0**

 Commits **1**

 Files changed **1**





larsyde commented just now

Owner



getting it done

  creating the todo list ...

c615885

Add more commits by pushing to the `my-test-branch` branch on `larsyde/my-new-test-repository`.



This branch has no conflicts with the base branch

Merging can be performed automatically.

 **Merge pull request**



You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

Demo still

creating the todo list, fixes #1 #2

 **Open** larsyde wants to merge 1 commit into master from my-test-branch

 Conversation 0

 Commits 1



 Files changed 1



larsyde commented a minute ago

Own

getting it done

  creating the todo list ...

Add more commits by pushing to the my-test-branch branch on larsyde/my-new-test-repository.



Merge pull request #2 from larsyde/my-test-branch


creating the todo list, fixes #1

 **Confirm merge**

Cancel

Demo still

creating the todo list, fixes #1 #2

 **Merged** larsyde merged 1 commit into `master` from `my-test-branch` just now

 Conversation 0

 Commits 1

 Files changed 1





larsyde commented a minute ago



Owner



getting it done

  creating the todo list ...

c615885

  larsyde merged commit 3bfb480 into `master` just now

Revert



Avoid bugs by automatically running your tests.


Continuous integration can help catch bugs by running your tests automatically.
Merge your code with confidence using one of our continuous integration providers.

[Learn more](#)



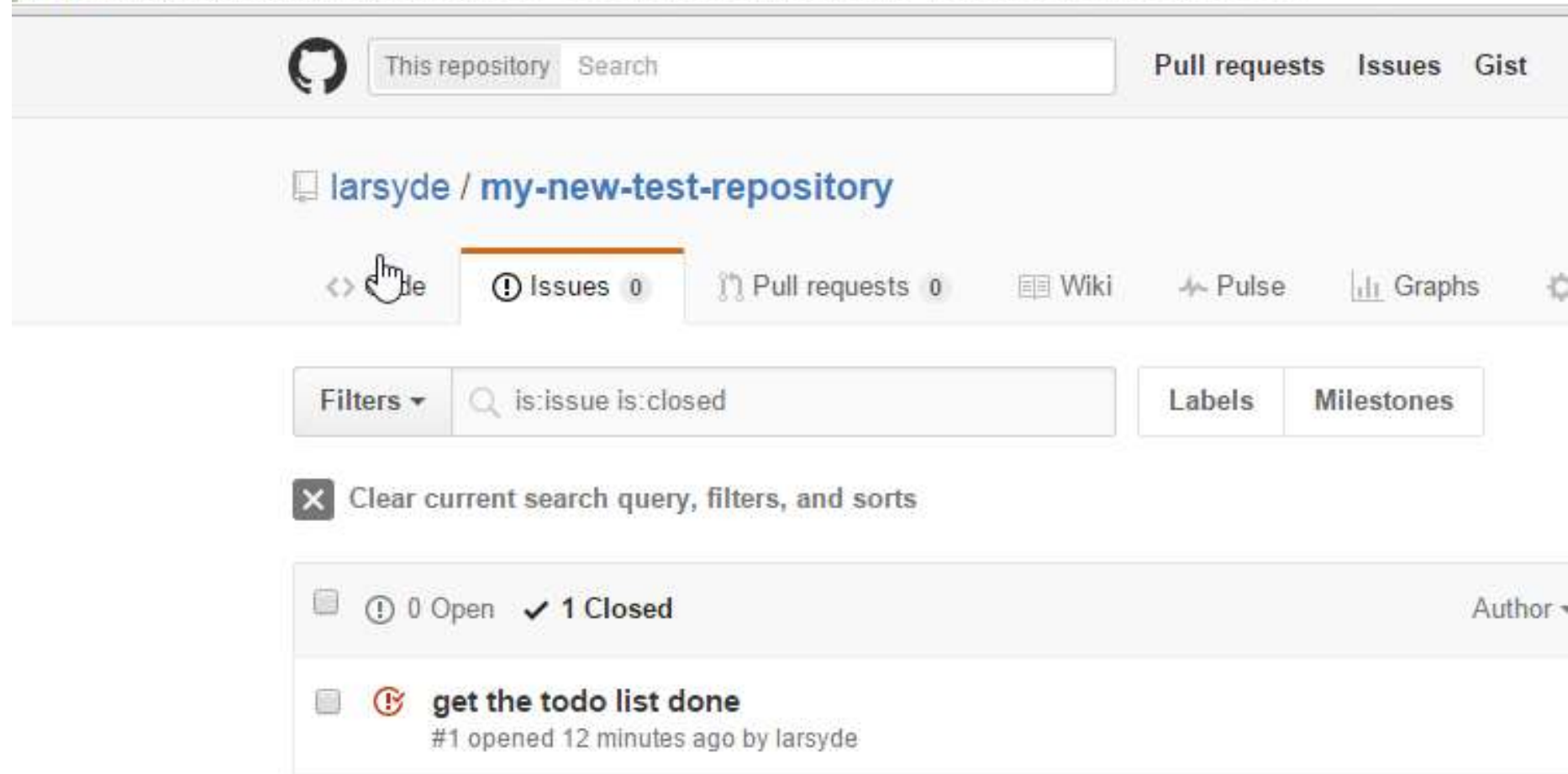
Pull request successfully merged and closed

You're all set—the `my-test-branch` branch can be safely deleted.

 Delete branch

Demo still

<https://github.com/larsyde/my-new-test-repository/issues?q=is%3Aissue+is%3Aclosed>





The screenshot shows the GitHub interface for the repository 'larsyde / my-new-test-repository'. The 'Issues' tab is selected, displaying a search bar with the query 'is:issue is:closed'. Below the search bar, there is a summary of issues: '0 Open' and '1 Closed'. A single issue is listed: 'get the todo list done', which was opened 12 minutes ago by the user 'larsyde'. The issue is marked as closed with a red 'X' icon. The interface also includes navigation links for 'Pull requests', 'Wiki', 'Pulse', and 'Graphs'.

Filters Labels Milestones

Clear current search query, filters, and sorts

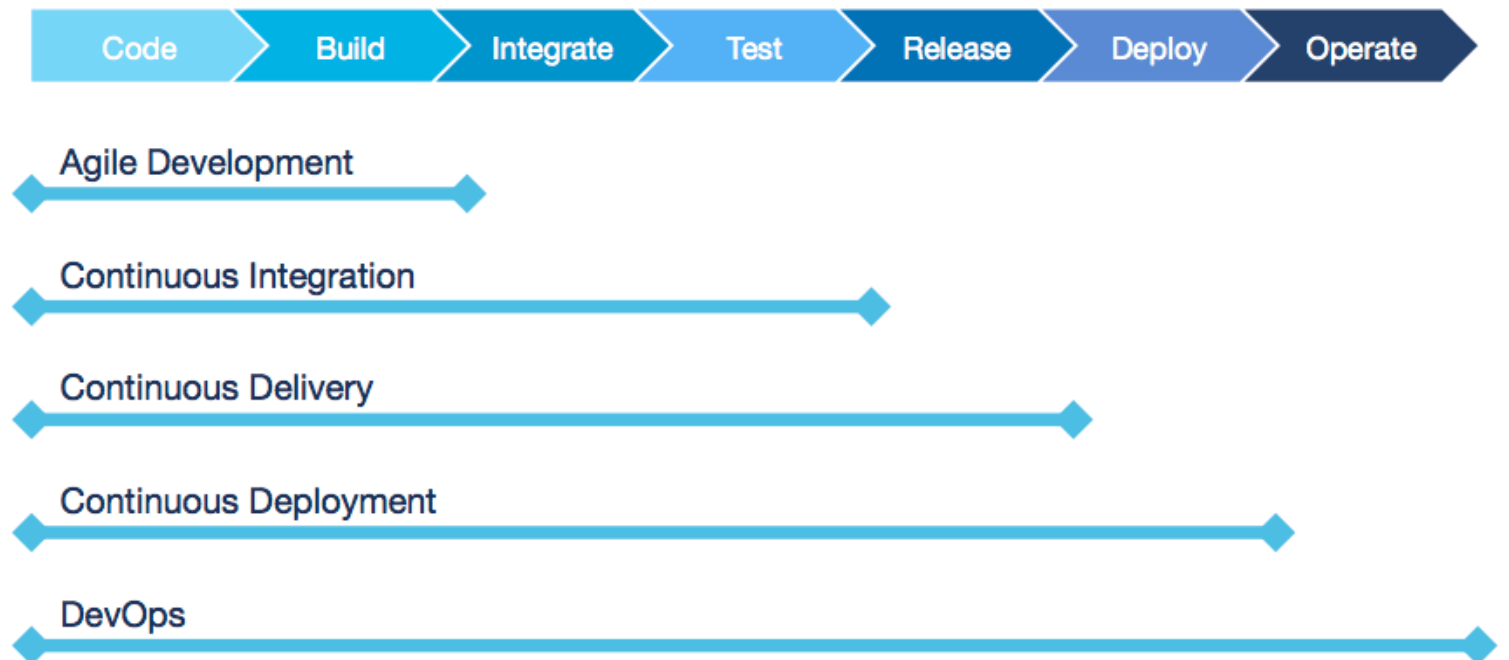
0 Open ✓ 1 Closed

☐  **get the todo list done**
#1 opened 12 minutes ago by larsyde

 **ProTip!** Notifv someone on an issue with a mention. I

Advanced topics - Github with CI / CD

- ▶ Continuous Integration (CI)
 - ▶ Integrate early, integrate often
 - ▶ Tight feedback loop
- ▶ Continuous Deployment / Delivery (CD) ~ DevOps
 - ▶ Repeatable process
 - ▶ End-to-end automation
 - ▶ Systems thinking
 - ▶ Amplify feedback loops
 - ▶ Culture of continual experimentation and learning



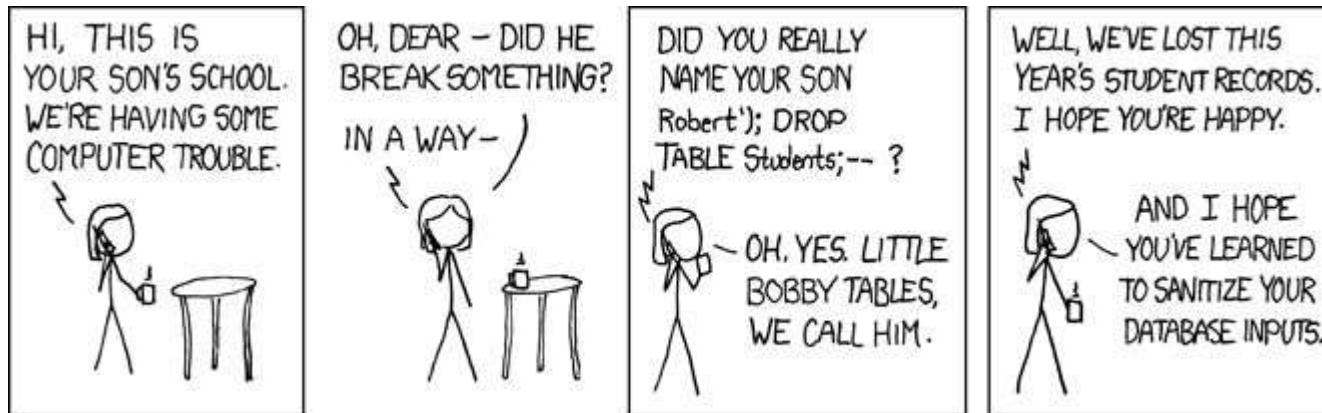
THIS IS GIT. IT TRACKS COLLABORATIVE WORK
ON PROJECTS THROUGH A BEAUTIFUL
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL
COMMANDS AND TYPE THEM TO SYNC UP.
IF YOU GET ERRORS, SAVE YOUR WORK
ELSEWHERE, DELETE THE PROJECT,
AND DOWNLOAD A FRESH COPY.



Issue tracking



Issue tracking

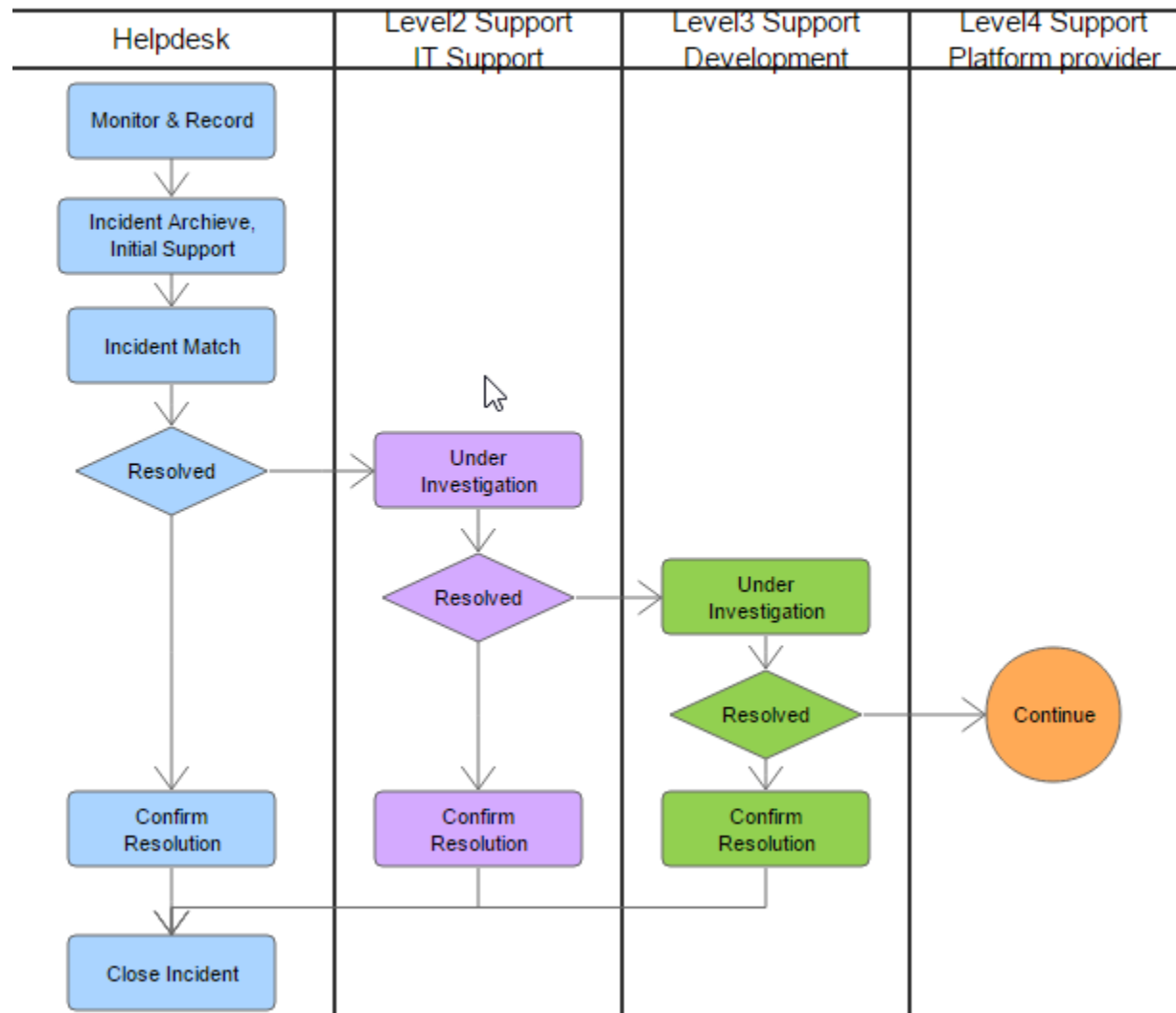
- ▶ What is it ?
 - ▶ Incident management = project management (ITIL) discipline with an ISO standard
 - ▶ A TODO list you keep on your device
 - ▶ A mental list you (may) remember before dev complete
 - ▶ A shared list (e.g spreadsheet) of description and metadata that is collaboratively maintained
 - ▶ A release notes document that follows the software
 - ▶ A tracking system that maintains a database of incidents / issues / bugs, relevant metadata, history information and secondary data (forensic data, resolution files)
 - ▶ A definitions game: what constitutes an issue, who fixes it, who's responsible ?

Issue tracking \geq bug tracking

- ▶ Defects may be in implementation (bugs) or elsewhere - requirements, design, documentation - as defects
- ▶ Why issue tracking ?
 - ▶ Capture
 - ▶ Record
 - ▶ Ensure accountability
 - ▶ Store resolution
 - ▶ Facilitate review

Incident management (ITIL)

Incident Management - ITIL V2



Incident management (ITIL) [14]

- ▶ ” An 'Incident' is any event which is not part of the standard operation of the service and which causes, or may cause, an interruption or a reduction of the quality of the service.”
- ▶ “The objective of Incident Management is to restore normal operations as quickly as possible with the least possible impact on either the business or the user, at a cost-effective price.”
- ▶ An ISO standard (ISO 20000)
- ▶ An exam (ITIL certification)
- ▶ A process



The TODO list

- ▶ Simple
- ▶ Easy to use and share
- ▶ Can be version-controlled
- ▶ Ships easily with the product as a release notes
- ▶ However...
- ▶ No workflow support
- ▶ No history
- ▶ No concurrency

Issue tracking systems [15]

- ▶ Open source (Bugzilla, Mantis, Redmine)
- ▶ Commercial (JIRA, ClearQuest, Trac)
- ▶ Integrated (Eclipse, TFS / Visual Studio, NetBeans)
- ▶ Auxiliary uses [18]
 - ▶ Knowledge repository
 - ▶ Communication and coordination hub
 - ▶ Communication channel
 - ▶ Context repository

Issue tracking system - Bugzilla

Bugzilla - Bug 2323 This is Bugzilla Last modified: 2009-01-06 16:32:48 PST

Home | New | Search | Find | Reports | My Requests | My Votes | Preferences | Administration | Help | Log out lukas@bugzilla.org

Bug List: (This bug is not in your last search results) [Show last search results](#)

Bug 2323 - This is Bugzilla (edit)

Status: REOPENED (edit)

Product: WorldControl

Component: WeatherControl

Version: 1.0

Platform: All All

Importance: P1 blocker (vote)

Target Milestone: ---

Assigned To: Max K-A (edit)

QA Contact: cluebot on irc.freenode.net (edit)

URL:

Whiteboard:

Keywords: KeyMe, KeyMe+

Depends on: 2324 2325 (edit)

Blocks:

[Show dependency tree / graph](#)

Reported: 2005-01-08 21:03 PST by Max K-A

Modified: 2009-01-06 16:32 PST (History)

CC List: ☐ Add me to CC list 2 users (edit)

Multi-Select: Option 1 Option 2

Drop Down: ---

Date Time:

Flags:

blocker ☐

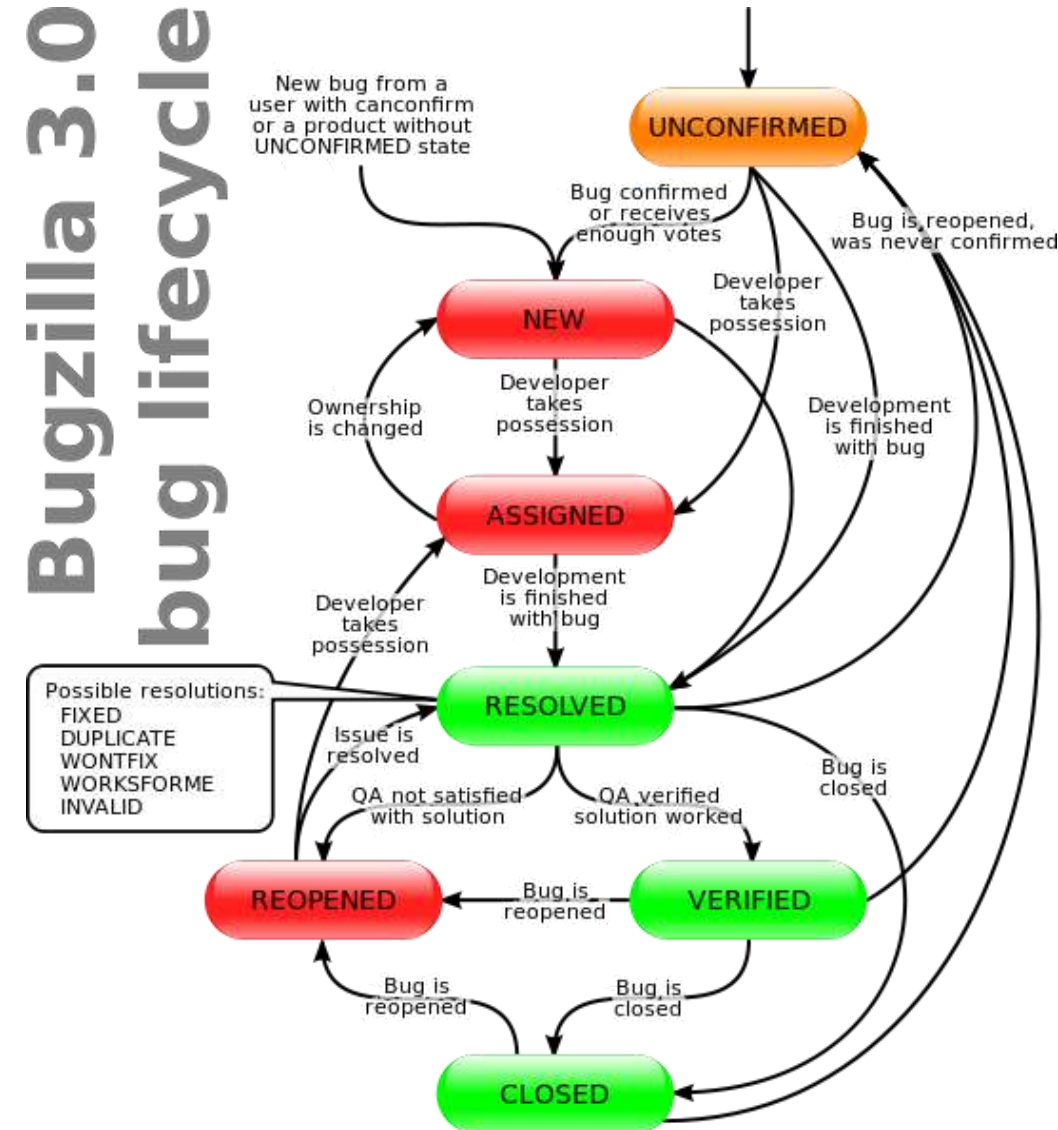
foo ☐

regression ☐

test ☐

Attachments

Bugzilla 3.0 bug lifecycle



Defect Triage Meeting / Bug Council

- ▶ What is triage ?
- ▶ Assemble stakeholders: development leads, managers, product owners, test leads, etc
- ▶ Ensure correct severity and priority for bug backlog
- ▶ Rinse and repeat until all is triaged



Communication



Small group communication theory [24][25]

► Systems theory

- “...groups are open systems, which are influenced by such independent variables as; openness to environment, interdependence, input variables, process variables, and output variables”

► Social exchange theory

- “[people] base the likeliness of developing a relationship with someone on the perceived possible outcomes. When these outcomes are perceived to be greater, we disclose more and develop a closer relationship with that person.”

► Symbolic convergence theory

- “In small groups, members develop private code words and signals that only those inside the group understand. When groups achieve symbolic convergence, they have a sense of community based on common experiences and understandings.”

► Structuration theory

- “[ST] views small groups as systems that both produce structures and are produced by structures. This means that group members follow particular rules in their interactions that produce some sort of outcome. That outcome eventually influences the group's future interactions.”

► Functional theory

- “Functional Perspective claims that there are four functions for effective decision making which include an analysis of the problem, goal setting, identification of alternatives, and an evaluation of positive and negative characteristics, all of which are equally important.”

Group development [26]

- ▶ Life cycle models:
 - ▶ Describe the process of change as the unfolding of a prescribed and linear sequence of stages following a program that is prefigured at the beginning of the cycle (decided within the group or imposed on it).
- ▶ Teleological models:
 - ▶ Describe change as a purposeful movement toward one or more goals, with adjustments based on feedback from the environment.
- ▶ Dialectical models:
 - ▶ Describe change as emerging from conflict between opposing entities and eventual synthesis leading to the next cycle of conflict
- ▶ Evolutionary models:
 - ▶ Describe change as emerging from a repeated cycle of variation, selection and retention and generally apply to change in a population rather than change within an entity over time.

Tuckman's model of group development

- ▶ Most popular team development model in agile
- ▶ Tuckman identifies four stages (and one additional stage in later versions)
 - ▶ Forming
 - ▶ “Groups initially concern themselves with orientation accomplished primarily through **testing**. Such testing serves to identify the boundaries of both interpersonal and task behaviors. Coincident with testing in the interpersonal realm is the establishment of dependency relationships with leaders, other group members, or pre-existing standards.”
 - ▶ Storming
 - ▶ The second point in the sequence is characterized by **conflict** and **polarization** around interpersonal issues, with concomitant emotional responding in the task sphere. These behaviors serve as **resistance** to group influence and task requirements”
 - ▶ Norming
 - ▶ “Resistance is overcome in the third stage in which **in-group feeling** and **cohesiveness** develop, new **standards evolve**, and new roles are adopted. In the task realm, intimate, personal opinions are expressed”
 - ▶ Performing
 - ▶ “Finally, the group attains the fourth and final stage in which interpersonal **structure** becomes the **tool** of task activities. Roles become **flexible** and **functional**, and group energy is channeled into the task. Structural issues have been resolved, and **structure** can now become **supportive** of task performance”

Tuckman visualized

Forming

Team acquaints and establishes ground rules. Formalities are preserved and members are treated as strangers.



Storming

Members start to communicate their feelings but still view themselves as individuals rather than part of the team. They resist control by group leaders and show hostility.



Norming

People feel part of the team and realize that they can achieve work if they accept other viewpoints.



Performing

The team works in an open and trusting atmosphere where flexibility is the key and hierarchy is of little importance.

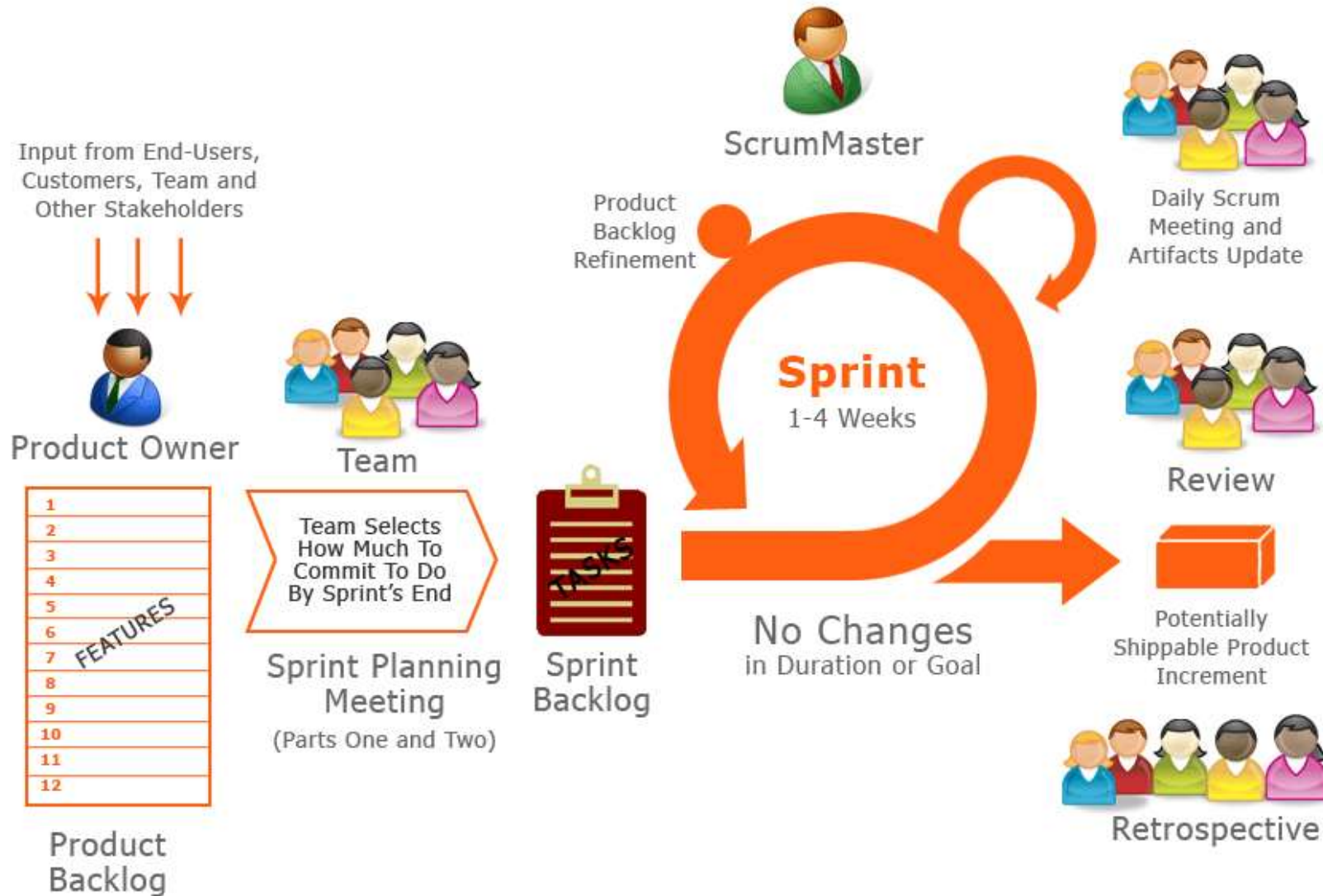


Adjourning

The team conducts an assessment of the year and implements a plan for transitioning roles and recognizing members' contributions.



Agile practices and communication



Structure: traditional vs agile

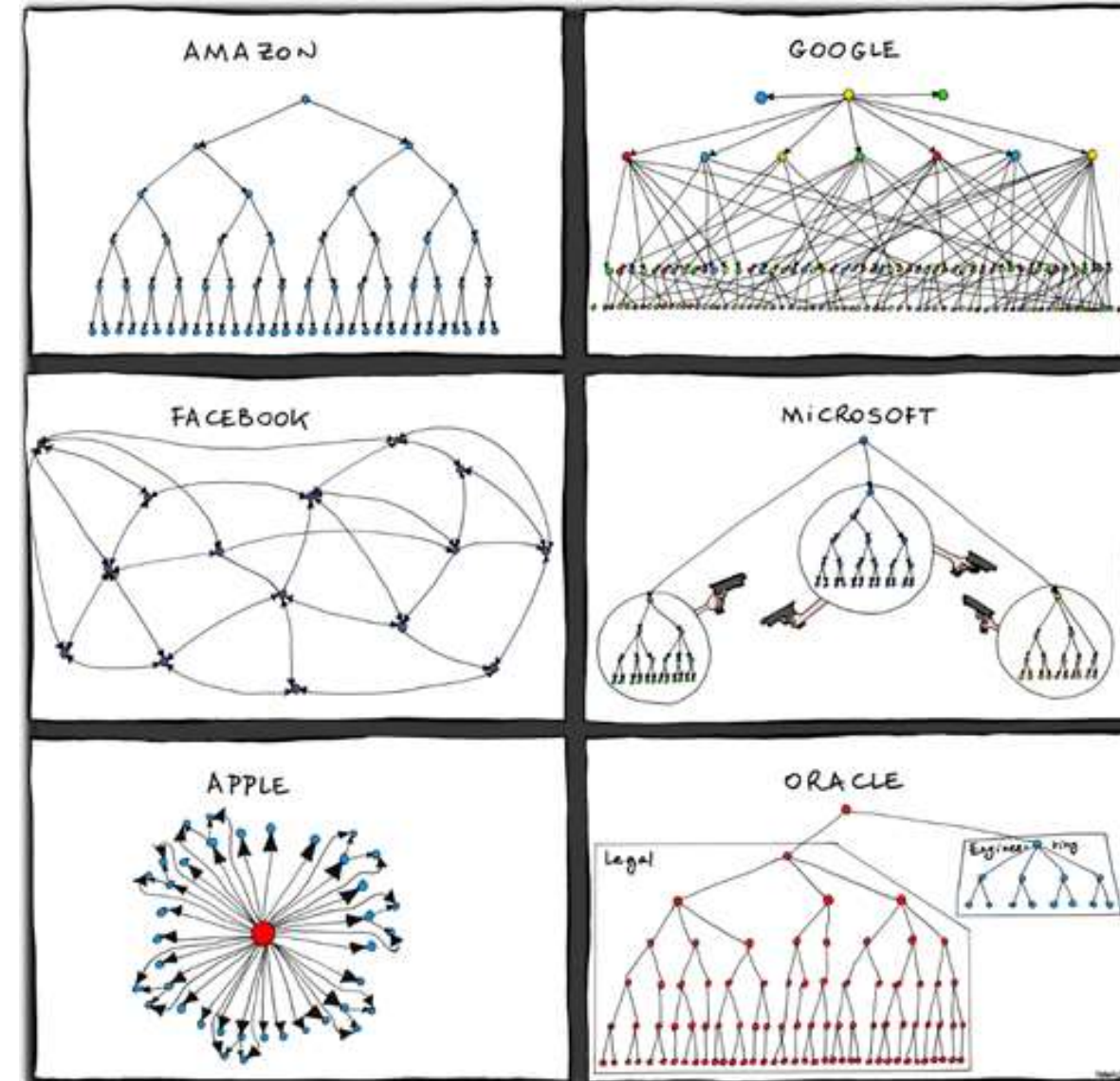
Practice	Waterfall	Iterative (hybrid)	Agile
Daily Standup	Daily/Weekly Status reports. PM calls for the status meeting at scheduled intervals	Daily/Weekly Status reports. PM calls for the status meeting at scheduled intervals.	Scrum Master facilitates the daily standup meeting to update: <ul style="list-style-type: none">• what we did?,• what we are going to do?• where we lag?
Status Report	Status report in prescribed template. More focus on Percentage done.	Status report in prescribed template. More focus on Percentage done.	Update on daily basis by logging hours spent & hours required to complete. Burndown/Burnup chart reflects the remaining hours required to complete
Planned vs Actual	Stick to baseline project plan	Stick to baseline iteration plan	Actual hours burnt vs hours required to complete

Barriers to communication

- ▶ Lack of unplanned contact [22]
 - ▶ Knowing who to contact about what,
 - ▶ Cost of initiating contact,
 - ▶ Ability to communicate effectively, and
 - ▶ Lack of trust, or willingness to communicate openly.

Organisation and technology as communication matrix

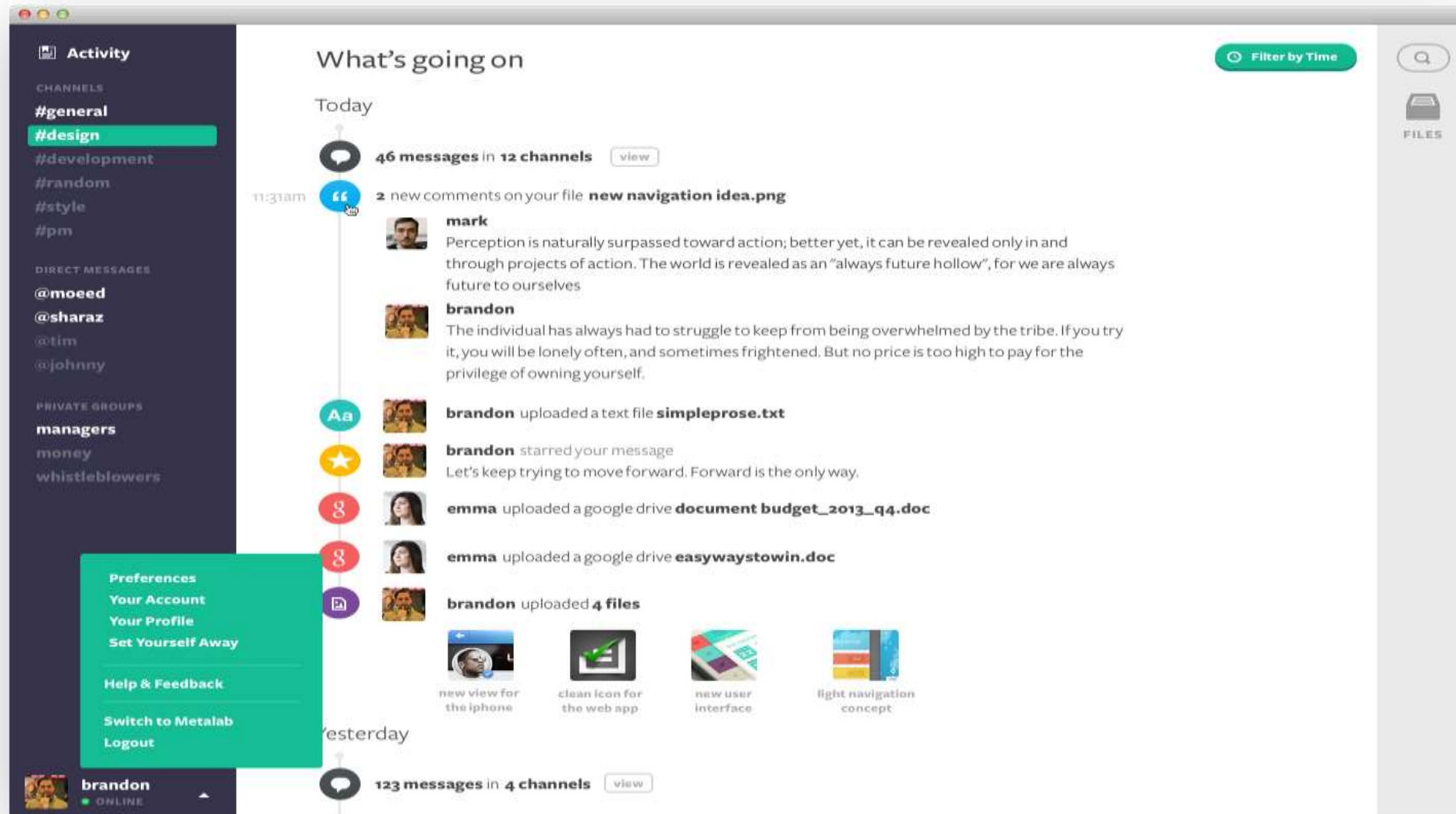
- ▶ Conway's law [21]
 - ▶ “organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations”
- ▶ Brook's law [22]
 - ▶ “if a project is late then adding more people to the development will slow the work down further”
 - ▶ “the addition of more people creates communications overhead”



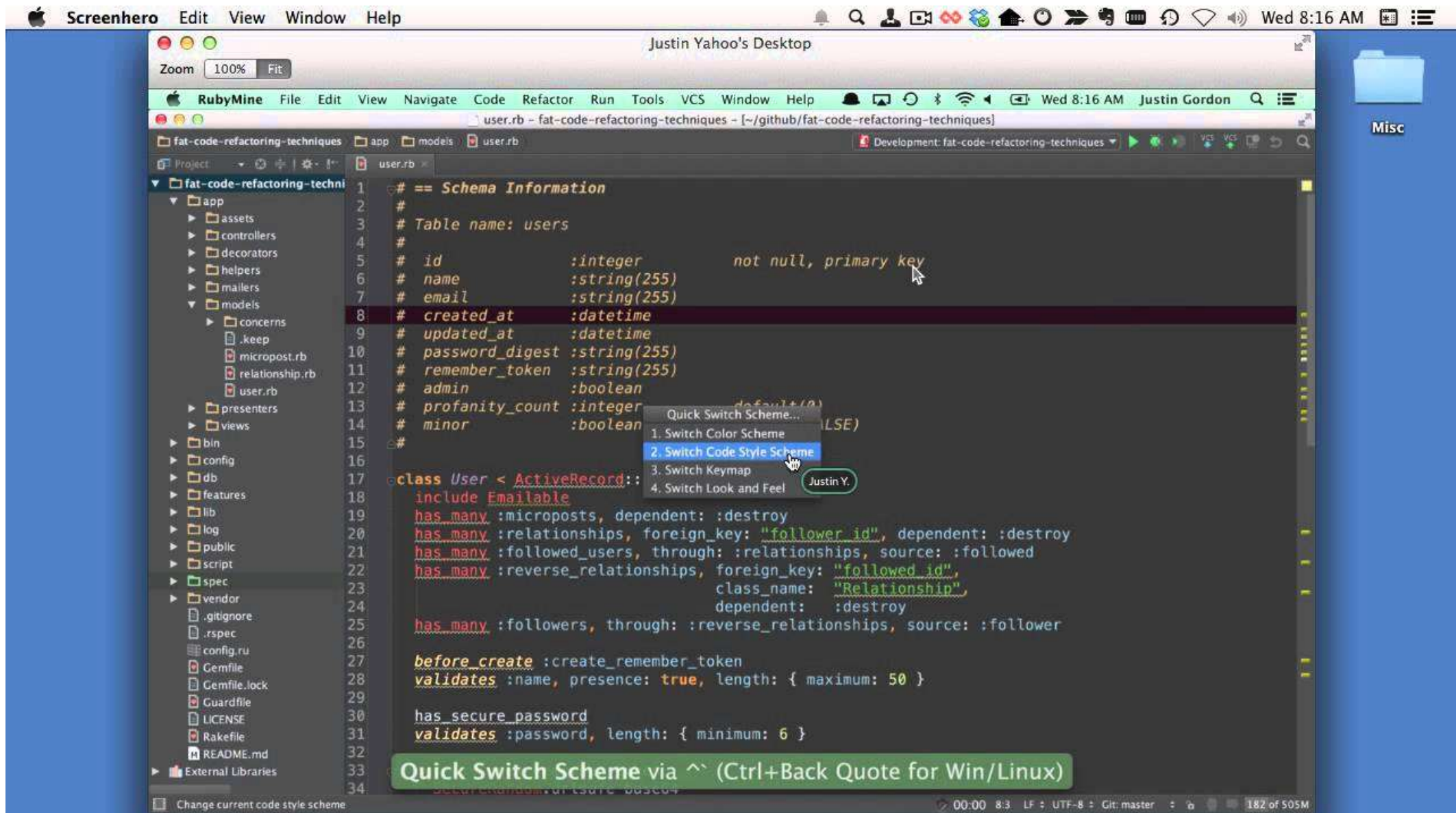
Communication tool support [19]

- ▶ A distributed VCS aids in meeting communication needs[5]
 - ▶ Improve cooperation, communicate feeling / sentiment, measure progress
 - ▶ Reduce uncertainty, act as a knowledge hub
 - ▶ Flow communication at the horizontal level of an organisation
- ▶ Groupware / CSCW applications
 - ▶ Document sharing (non-VCS'ed) : Quip, Dropbox, Screen Hero
 - ▶ Communications : Skype, Mail clients, Slack, Glip, HipChat, MatterMost
 - ▶ Project managerial : Trello, shared calendars, wikis, group sites, spread-sheets

Slack



Screen Hero



Trello

Boards


The Great Kitchen Redesign

Taco's Organization

Public

Ideas

Get a new window valence to match the cabinet colors



Install pot rack over the island

1 2/3

Replace drawer knobs with antique ones

4


Add a card...

To Do

Adjust water pressure of the sink

1 vote 0/4 Nov 10, 2013


Remove old refrigerator and stove




Install new sink

1 0/10 Nov 4, 2013

Install new flooring





Buy paint for cabinets

1

Add a card...


Doing

Pick countertop colors

Nov 27, 2013

Buy new kitchen cart

1



Design new kitchen space


1 vote 2

Add a card...

Done!

Call contractor

1/2




Pick faucet to match new sink

1 vote 2

Add a card...

Menu

Members



Add Members...

Activity

Adam Simms changed the background of this board.
Jul 7 at 2:06 pm

Adam Simms changed the background of this board.
Jul 7 at 2:05 pm

Tracey Marlow moved Pick faucet to match new sink from Doing to Done!.
Jun 23 at 2:43 pm

Adam Simms renamed this board (from Remodel the Kitchen). Jun 23 at 2:30 pm

Tracey Marlow joined Pick faucet to match new sink.
Jun 23 at 1:41 pm

Tracey Marlow joined Remove old refrigerator and stove. Jun 23 at 1:40 pm

Tracey Marlow joined Replace drawer knobs with

Skype



Quip



LOOK THIS NEW AGILE THING:
TO DEAL WITH
UNPREDICTABLE EVENTS AND
THINGS WE CANNOT CONTROL
IN OUR PROJECTS



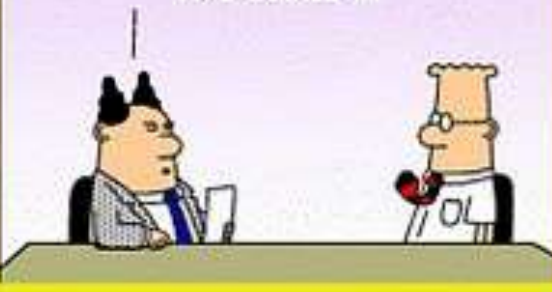
Dilbert characters Scott Adams Inc.

WE CAN PRIORITIZE, REDUCE
THE SCOPE, CHANGE
REQUIREMENTS AT ANY TIME
AND INCREASE THE CHANCES
OF SUCCESS OF THE PROJECT



Punch your own at <http://dilbert.com>

LOOK, THIS IS YOUR NEW
PROJECT, WITH FIXED
DEADLINE, FIXED SCOPE AND
FIXED QUALITY: YOU CAN BE
"AGILE" INSIDE THIS
TRIANGLE !!!



References

1) Distributed vs centralized

- ▶ <http://www.drdoobs.com/architecture-and-design/distributed-vcs-understanding-the-paradi/240159530>

2) Communication Tools for Distributed Software Development Teams

- ▶ https://www.rti.org/pubs/rti_cpr07_virtualteam.pdf

3) Communication patterns and practices in software development networks

- ▶ <http://www.soberit.hut.fi/veto/english/Julkaisut/ComPatterns.pdf>

4) Making sense of revision-control systems

- ▶ <http://cacm.acm.org/magazines/2009/9/38901-making-sense-of-revision-control-systems/fulltext>

5) The role of Distributed Version Control Systems in team communication and learning

- ▶ https://gupea.ub.gu.se/bitstream/2077/36326/1/gupea_2077_36326_1.pdf

6) Eclipse community survey 2014

- ▶ <http://www.slideshare.net/IanSkerrett/eclipse-community-survey-2014>

7) Revision Control System (documentation)

- ▶ <https://www.gnu.org/software/rcs/manual/rcs.html>

8) SVN "Red book" manual

- ▶ <http://svnbook.red-bean.com/en/1.7/svn-book.pdf>

9) Git "Pro git" manual

- ▶ <https://progit2.s3.amazonaws.com/en/2016-01-21-82fa5/progit-en.998.pdf>

References

10) Version control terminology

- ▶ https://en.wikipedia.org/wiki/Version_control

11) Version control by example

- ▶ http://ericsink.com/vcbe/vcbe_a4_lo.pdf

12) A visual guide to version control

- ▶ <http://betterexplained.com/articles/a-visual-guide-to-version-control/>

13) Continuous integration

- ▶ <http://martinfowler.com/articles/continuousIntegration.html>

14) Incident management (ITIL)

- ▶ http://itlibrary.org/index.php?page=Incident_Management

15) Comparison of issue tracking systems

- ▶ https://en.wikipedia.org/wiki/Comparison_of_issue-tracking_systems

16) ITSM best practices

- ▶ <http://www.itsm.info/ITSM%20Problem%20and%20Incident%20Best%20Practices.pdf>

17) Issue Tracking Systems

- ▶ http://is.muni.cz/th/60778/fi_m/thesis.pdf

References

18) The Social Nature of Issue Tracking in Software Engineering

▶ http://lsmr.org/docs/bertram_msc_2009.pdf

19) Communication Tools for Distributed Software Development Teams

▶ https://www.rti.org/pubs/rti_cpr07_virtualteam.pdf

20) Communication in Agile Teams

▶ <https://7bsp1018.wikispaces.com/Communication+in+Agile+Teams>

21) Conway's law

▶ https://en.wikipedia.org/wiki/Conway%27s_law

22) Splitting the Organization and Integrating the Code: Conway's Law Revisited

▶ https://www.cs.cmu.edu/afs/cs.cmu.edu/Web/People/jdh/collaboratory/research_papers/ICSE99.pdf

23) A Theory of Shared Understanding for Software Organizations

▶ https://tspace.library.utoronto.ca/bitstream/1807/26150/6/ArandaGarcia_Jorge_201011_PhD_thesis.pdf

24) Small group communication theory

▶ https://en.wikipedia.org/wiki/Communication_in_small_groups

25) Small group communication context

▶ <http://www.uky.edu/~drlane/capstone/group/>

26) Group development

▶ https://en.wikipedia.org/wiki/Group_development