Startup Engineering

Course and Project Information

Sung Kim

What are your expectations?

AWS, AngularJS등 최근 유행인 기술들을 익히고 싶습니다.
1/12/2016 1:35 PM View respondent's answers

여러가지 기술들을 배우고싶어 참여했습니다.
1/6/2016 5:43 PM View respondent's answers

프로그램이 설계 및 기획되고 또 제작되는 일련의 과정들을 경험해보고 싶었습니다. 그런 경험을 해봤던 적이 없었을 뿐만 아니라, 프로그래밍 기초과목을 수강하면서 다소 힘든 점들이 있었지만, 게임 구현 프로젝트 때는 프로그래밍이 생각보다 재밌는 분야라는 것을 깨달을 수 있었습니다. 더불어 저의 진로에 대해서도 고민을 해볼 수 있는 시간일 것 같습니다.
1/6/2016 5:02 PM View respondent's answers

여러가지공부를해보고싶어서입니다
1/6/2016 5:02 PM View respondent's answers

전공에 관한 다양한 분야에 대해서 배우고싶고 지난학기때 웹어플리케이션 과목을 수강하였는데 재미가있어서 더 심화된학습을 하고자 참여하게됬습니다.요새 아마존 웹클라우드 서비스가 뜨고있는데 한번 배워보고 싶습니다.
1/6/2016 4:30 PM View respondent's answers

웹페이지 개발을 배우기 위해 참여하게 되었습니다.
1/6/2016 4:27 PM View respondent's answers

What are your expectations?

AWS 관련 지식 습득

1/6/2016 4:19 PM View respondent's answers

일단 정확히 어떤 프로그램인지는 잘 모르겠는데....친구가 하자고 해서 했어요! 뭐, 그것보다 중요한 것은 내가 이 프로그램을 재미있어보인다고 생각해서겠지만. 배우고 싶은 것은 음..AWS의 핵심 기능들과 운용 테크닉?

1/6/2016 4:01 PM View respondent's answers

새로운 툴이나 전반적인 소프트웨어 개발 프로세스 등과 같이 기존에 몰랐던 다양한 정보를 알고 싶은 마음 반, 잘 배워놓으면 지금 인턴 업무에도 도움이 되고 앞으로의 나에게도 도움이 될 것 같다는 마음 반으로 신청했습니다. AWS, Git, 웹 및 모바일 어플리케이션에 대해 구체적으로 배워보고 싶습니다.

1/6/2016 3:58 PM View respondent's answers

웹서비스를 하는데 필요한 모든 것을 체험해볼 수 있다고 해서.

1/6/2016 3:53 PM View respondent's answers

웹에 관련된 많은 것들을 배우기 위해

1/6/2016 3:52 PM View respondent's answers

웹어플리케이션 수업을 들어보고 웹이 재밌었고 더욱 공부하고싶었습니다.

1/6/2016 12:55 PM View respondent's answers

What are your expectations?

기획부터 제작 출시까지 모든 과정을 배우고 싶어서

1/5/2016 10:09 PM View respondent's answers

참여한 동기는 제가 컴공 이긴 한대 아직 실력이 많이 부족하고 다른 실력있는 사람들을 보면서 많은 것을 배우고자 참여하게 되었습니다.

1/5/2016 10:02 PM View respondent's answers

MEAN stack을 이용한 웹앱 제작

1/5/2016 9:40 PM View respondent's answers

만들고자 하는 웹서비스가 있었는데 그 웹서비스를 만들기 위해서 Ruby on Rails와 HTML, CSS, Javascript등을 모두 동기와 공부를 하다보니 누군가에게 배우면 좀 더 이러한 도구들을 잘 활용할 수 있을것 같다는 생각이 들었고, 그러는 중에 이번 해커톤을 만나게 되어 참여하게 되었습니다.

1/5/2016 9:33 PM View respondent's answers

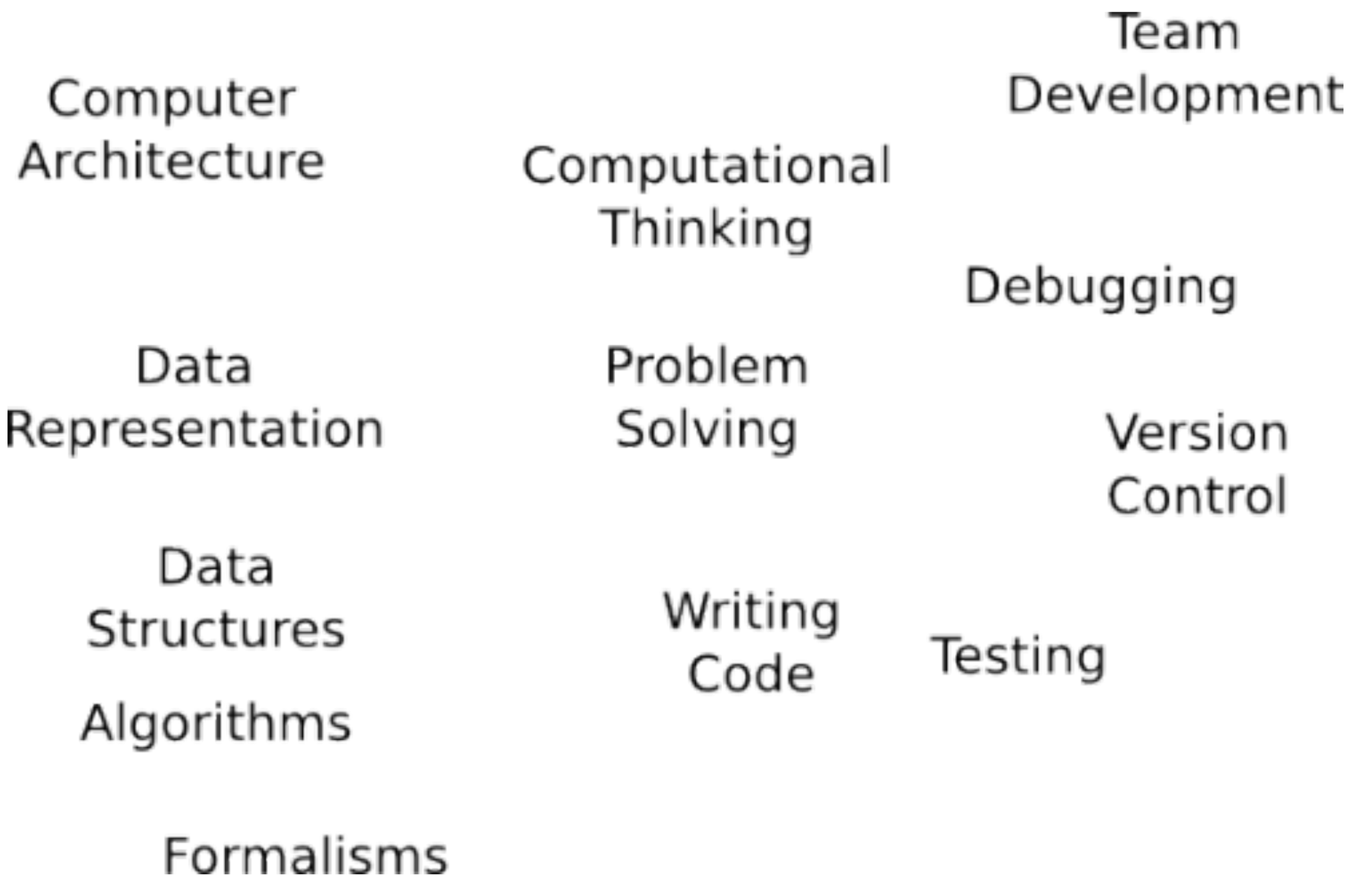
고급 웹 기술을 배워서 한층 더 성장된 프로그래머로써 사회에 나가기 위해 참여하게 되었습니다.

1/5/2016 9:29 PM View respondent's answers

여러가지 아이디어들을 웹과 앱으로 서비스하고 구현하는 방법, 노하우, 스킬을 배우기 위해서 참여하였습니다.

1/5/2016 9:25 PM View respondent's answers

Computer Science



https://academiccomputing.wordpress.com/2012/06/01/what-is-computer-science-what-is-programming/

http://eclnz.com/ecllabs

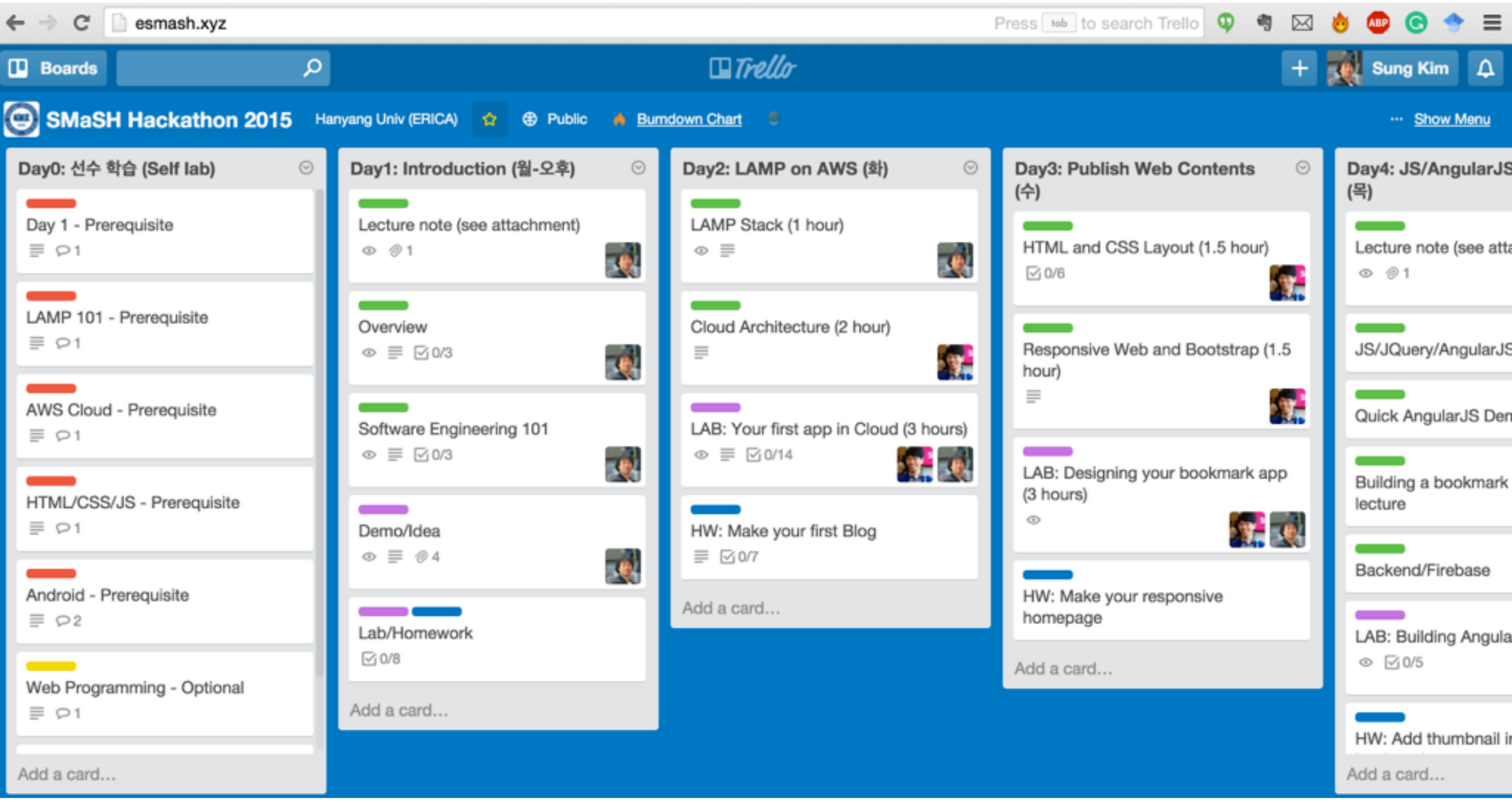
So what we do?

- From ideas to products
  - in a reasonable period (a week)
  - with reasonable SE methodologies

Topics

- Full stack web/mobile app
  - AWS/Unix/HTML/AngularJS/Android (Java)
  - Git/Github
  - Development processes
  - Testing/Debugging
  - Continus Integration/monitoring

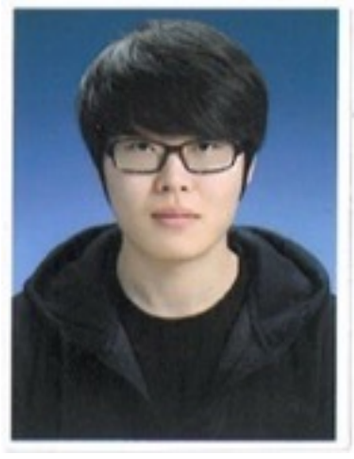
Class web page: **eSmash.xyz**



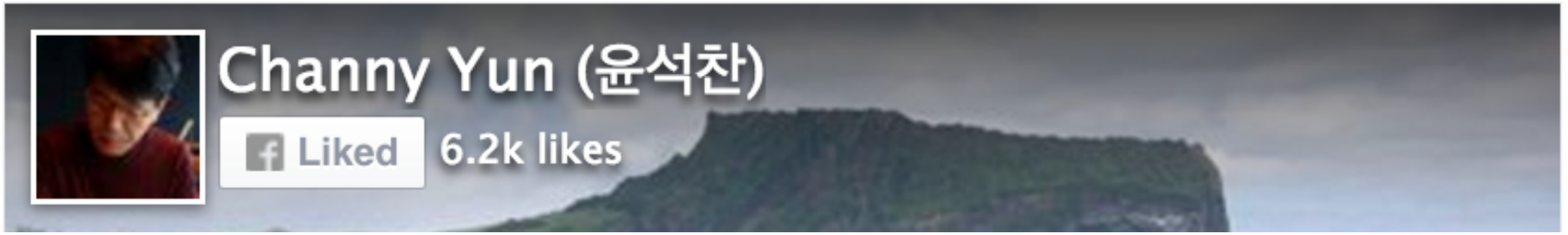
Meet Teaching Team

- 도경구 교수님, Scott Lee 교수님, 김광 교수님

- Instructors:윤석찬, 허광남, 김성훈



- TAs: 이정훈, 권기원, 김가연, 김진대



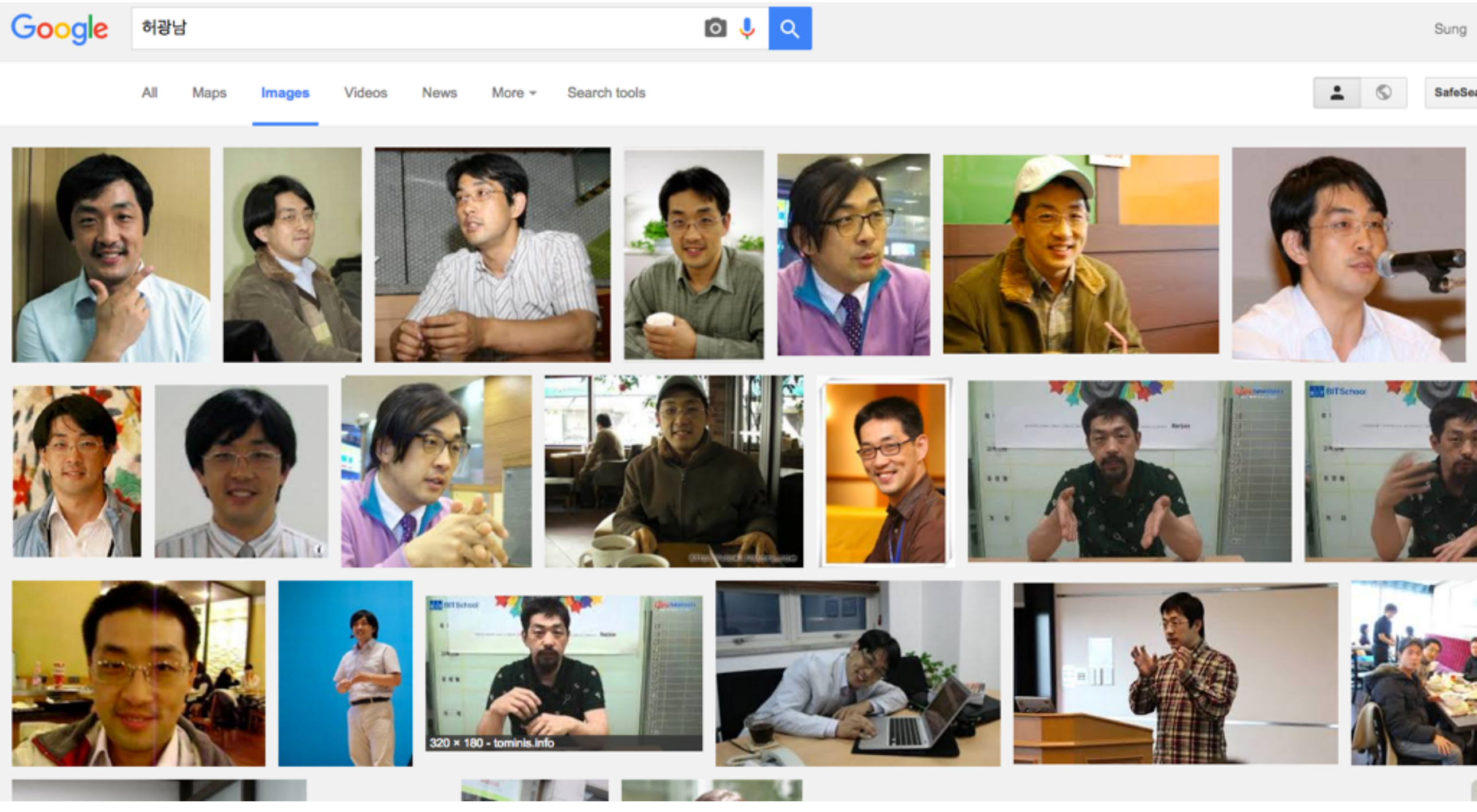
Channy Yun

Open Web and Cloud Evangelist in Korea

Channy works for Amazon Web Services as a Technology Evangelist for helping developers adoption of cloud computing in Korea. He has promoted Open APIs & Web standards as a developer evangelist in Daum.net and leader of Web Standards Korea, Open-source software as a professor of Jeju National University and a lead of Mozilla Korean Community.

He was chosen in one of top 20 powerful voices of Open-source software and well-known tech writer of his Channy's Blog with over 64K subscribers, 35k Twitter followers, 9k G+ follower and 6k Facebook fans. Also he helped developer's relationship as a founder of BarCamp Seoul, WebAppsCon, and DevOn.

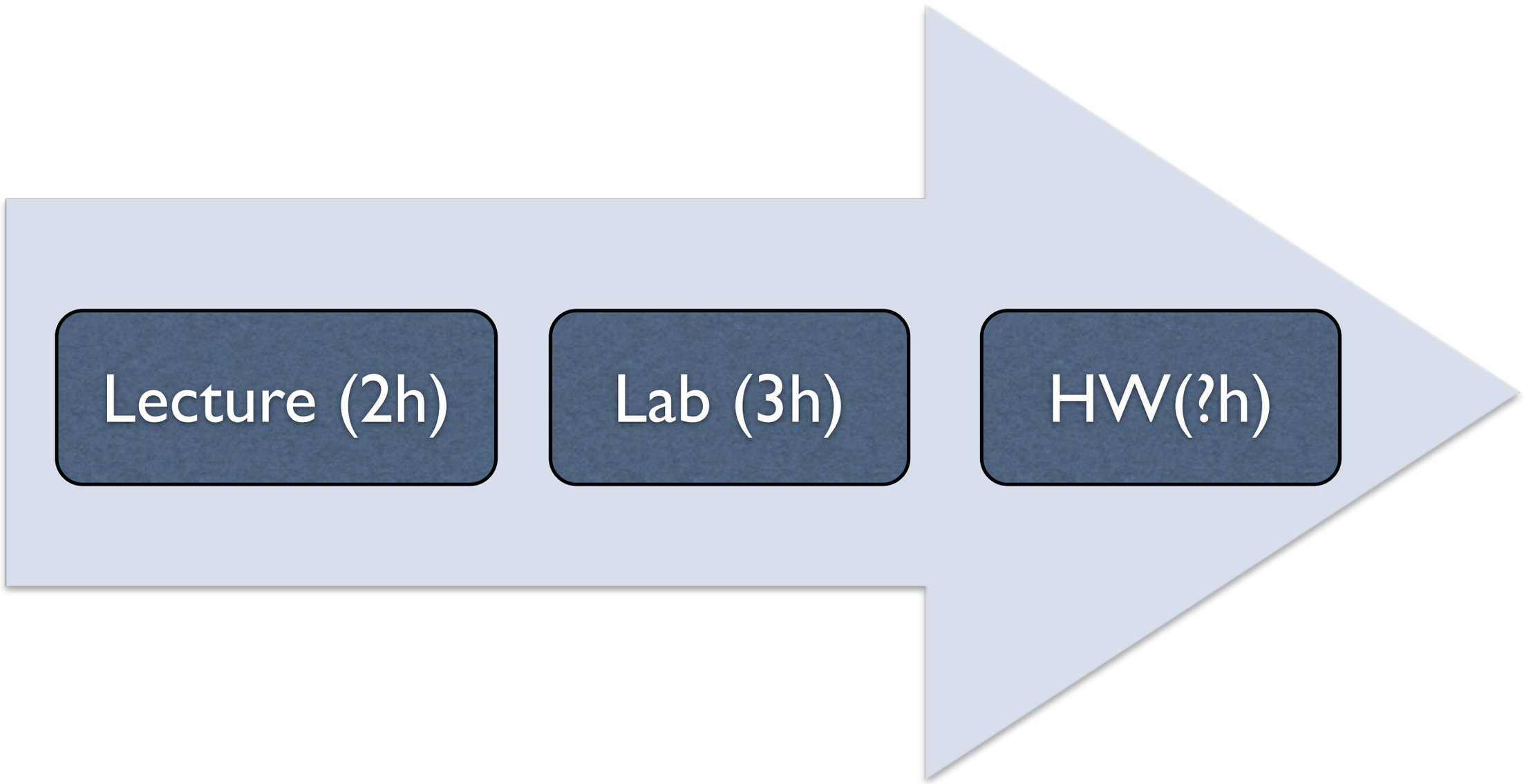
허광남


Google
허광남
Sung
All
Maps
Images
Videos
News
More
Search tools
SafeSea
320 × 180 - tominis.info

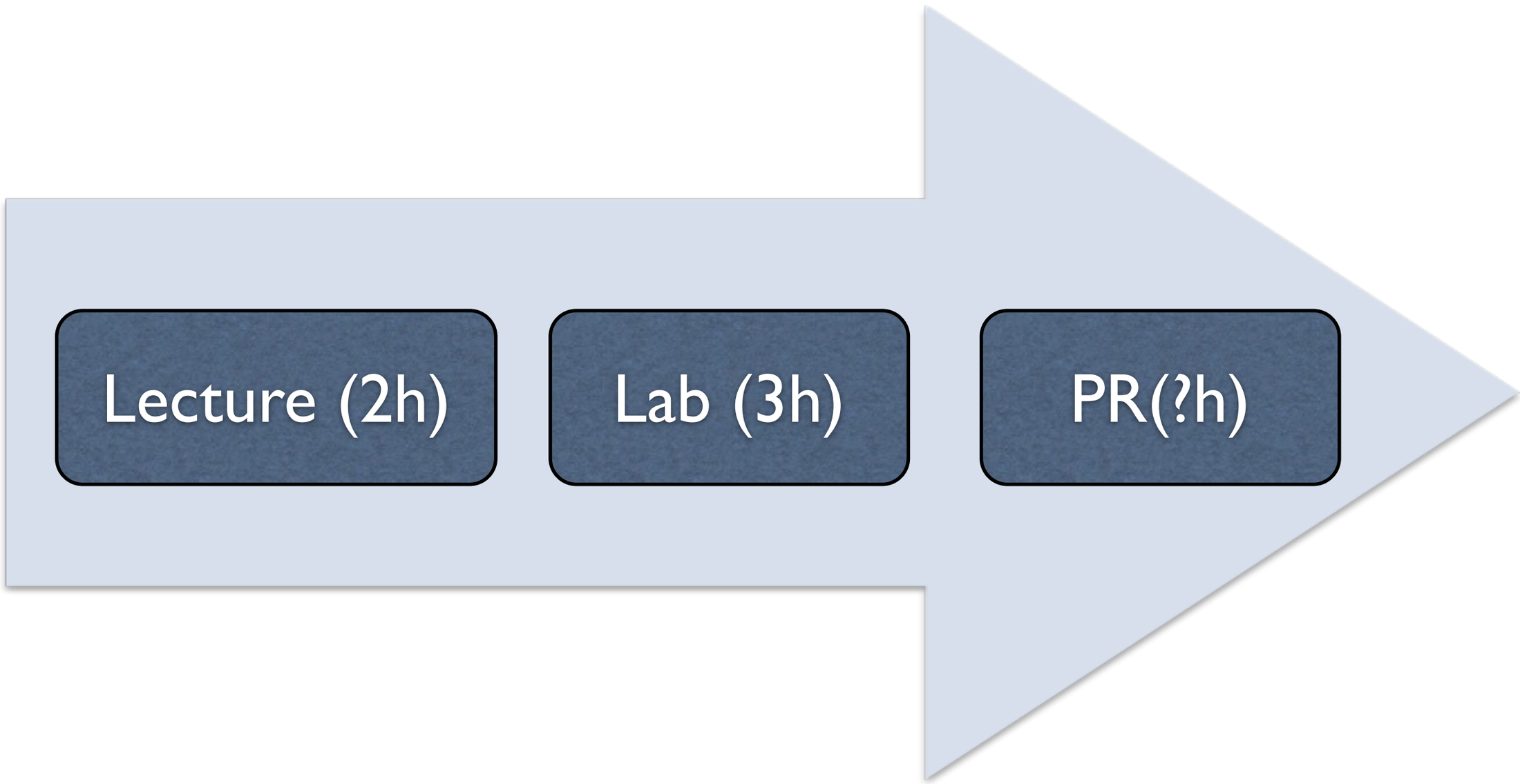
Meet Sung Kim

- 1995 - developed a Korean Search Engine
- 1996-2000 - created an Internet company, worked as CTO
- 2001-2006 - PHD study at University of California, Santa Cruz
- 2006-2008 - worked as a postdoc at MIT
- 2009- Associate Prof. at HKUST

Week 1: Lectures/Labs/HWs

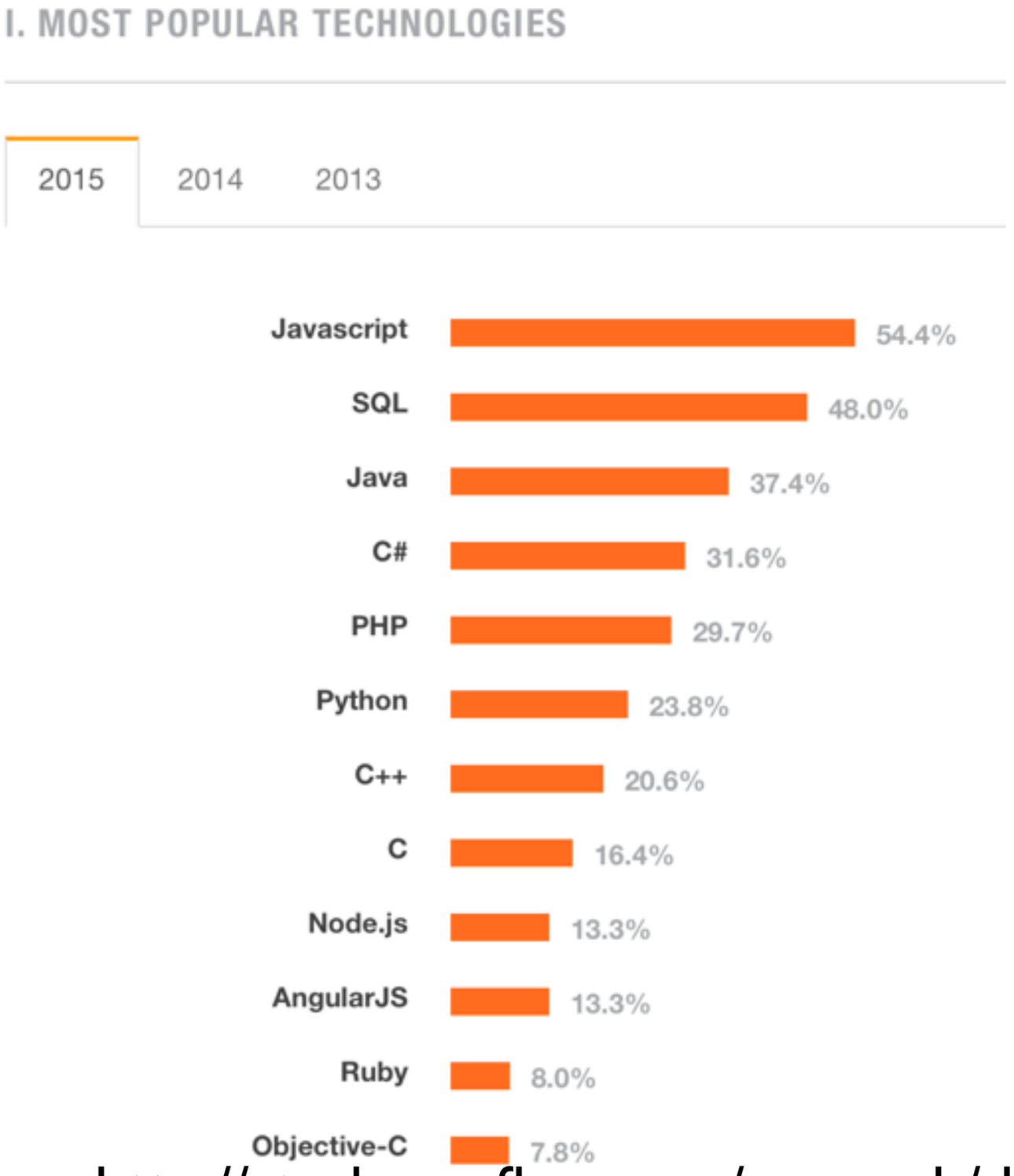


Week 2: Lectures/Labs/Team Projects



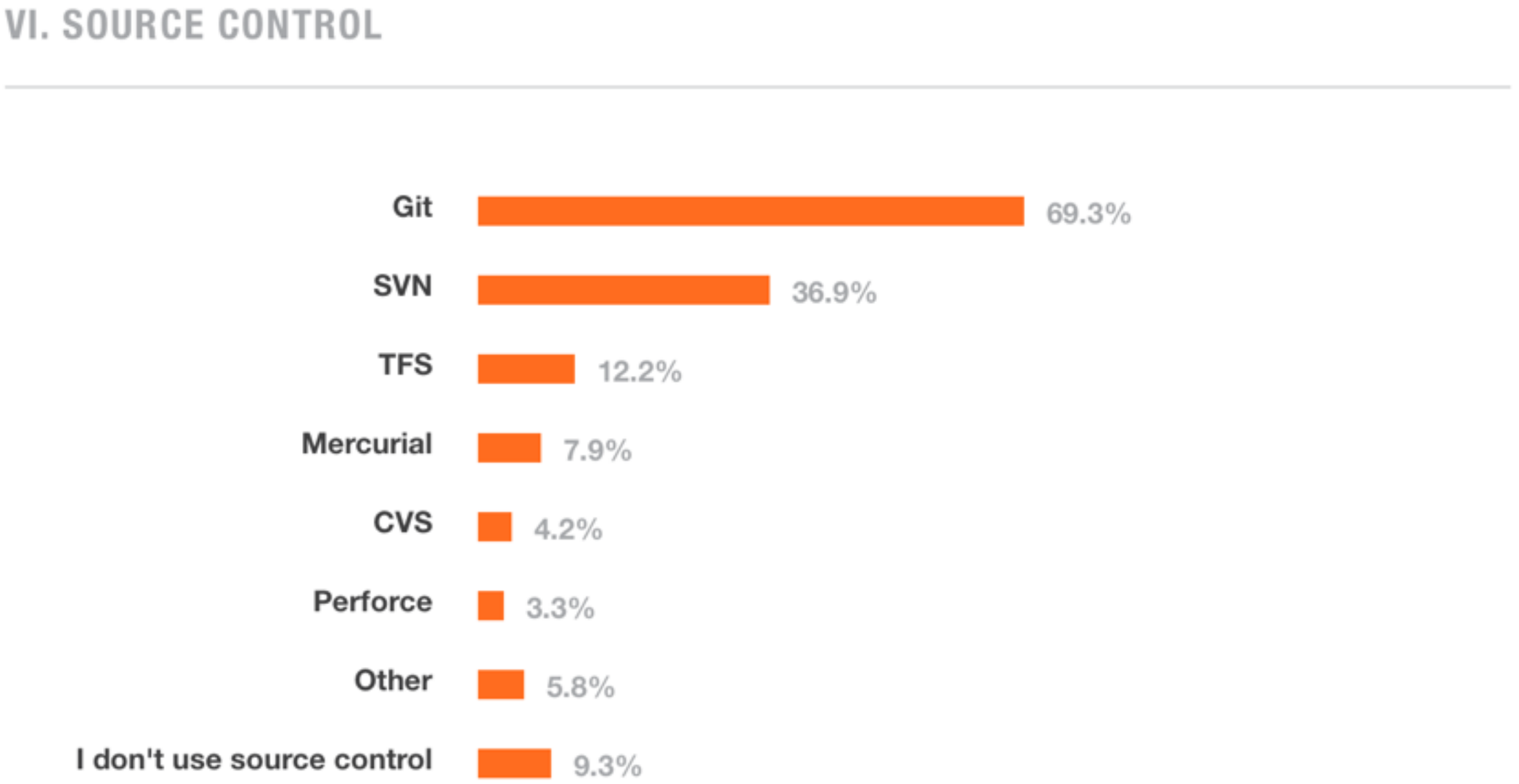
Key Technologies

StackOverflow developer surgery 2015 (26,086 people from 157 countries)



http://stackoverflow.com/research/developer-survey-2015

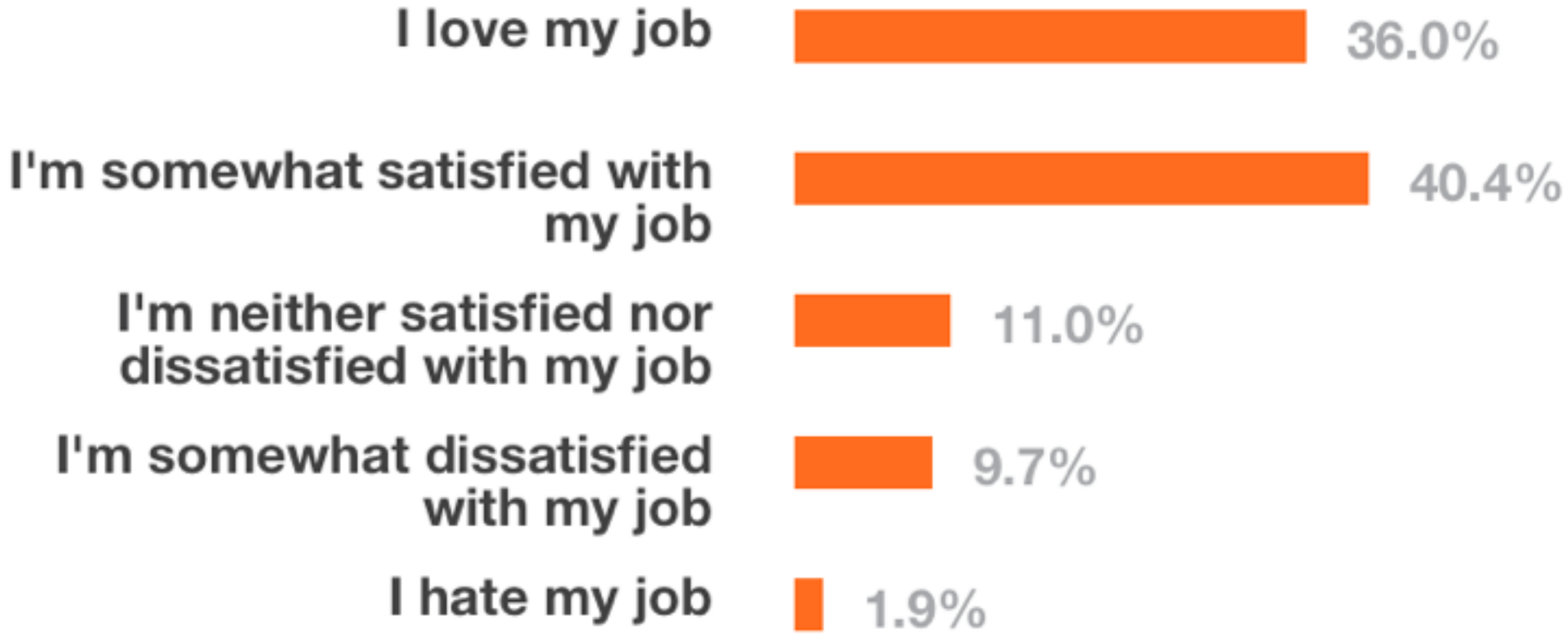
StackOverflow developer surgery 2015 (26,086 people from 157 countries)



http://stackoverflow.com/research/developer-survey-2015

StackOverflow developer surgery 2015 (26,086 people from 157 countries)

XI. JOB SATISFACTION



http://stackoverflow.com/research/developer-survey-2015

So what we do?

- From ideas to products
  - in a reasonable period (a week)
  - with reasonable **SE** methodologies

Why Software Engineering?

Software is Everywhere!

http://www.tuleap.com/?q=solutions/challenge/industrializing-software-development

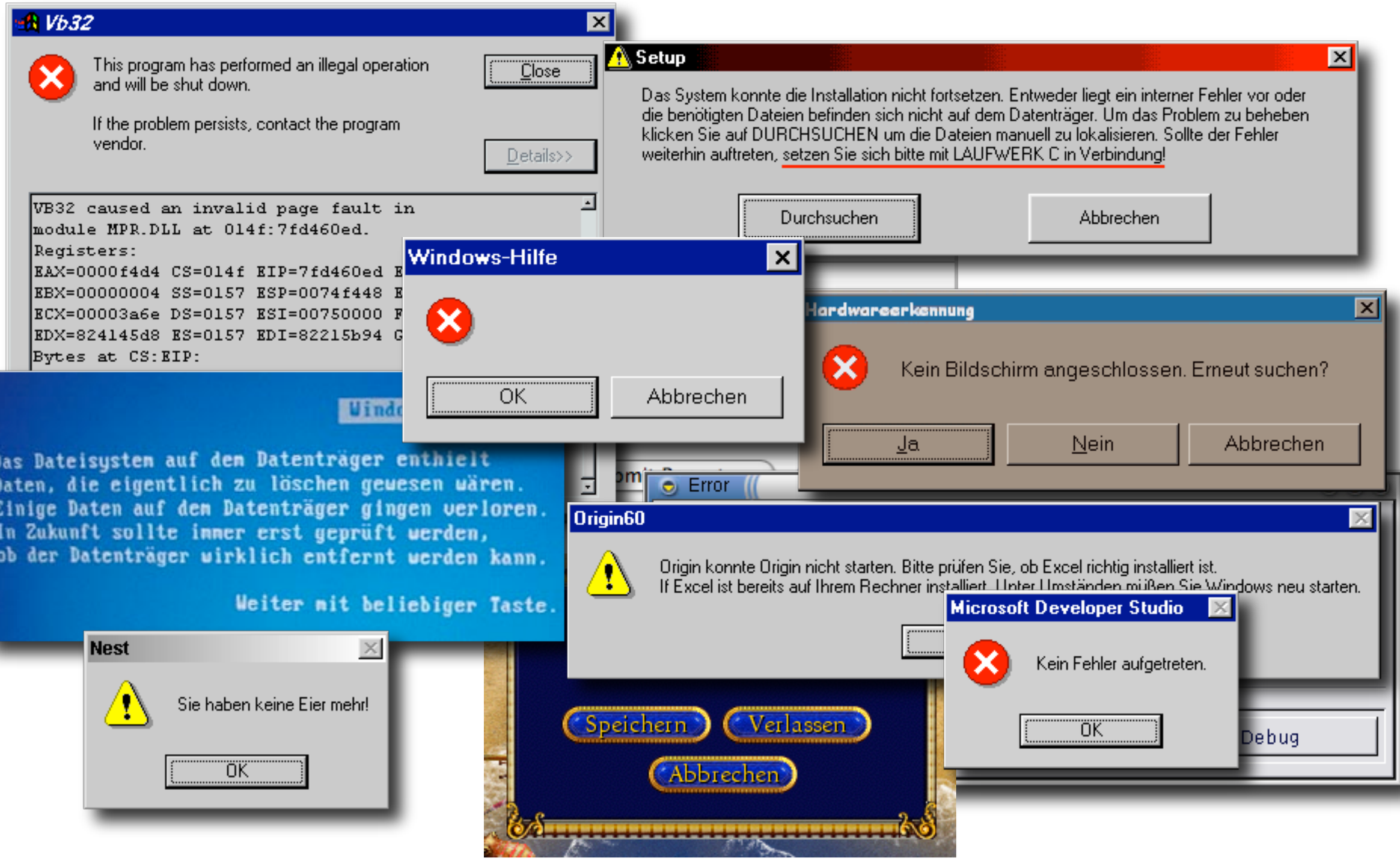
Software is Everywhere!

"Software is eating the world*"

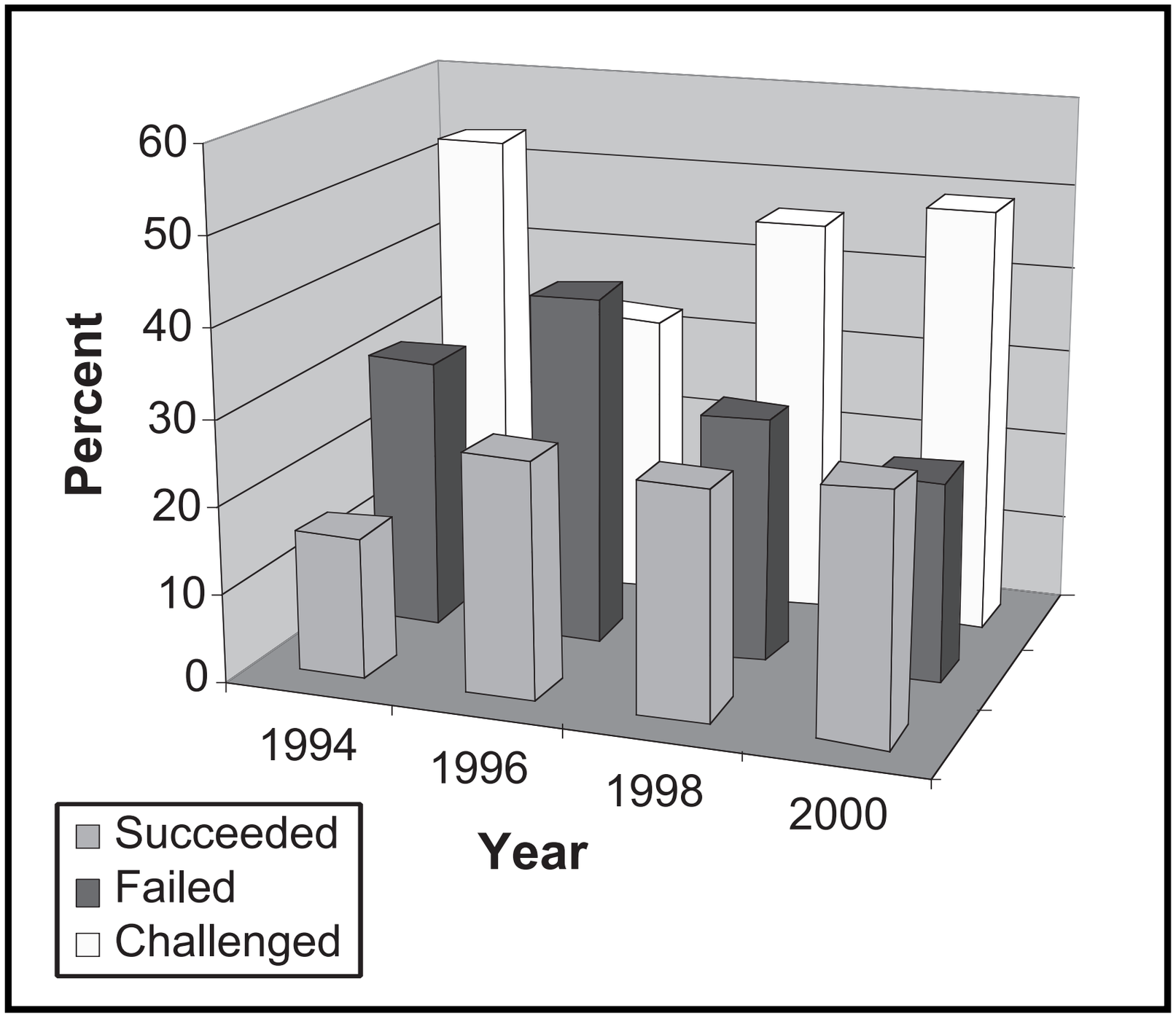
* quote by Andreessen in WSJ

http://www.tuleap.com/?q=solutions/challenge/industrializing-software-development

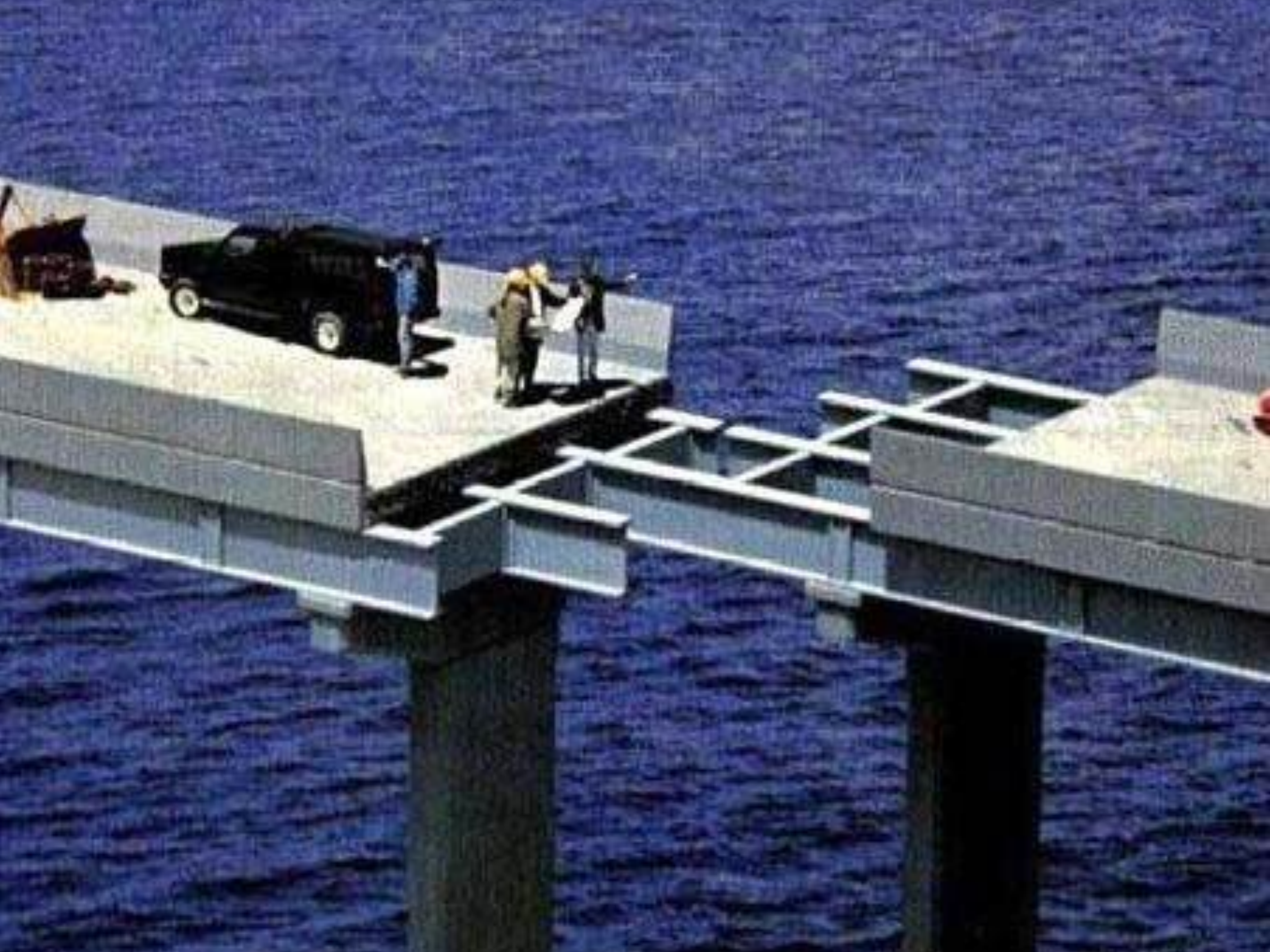
But software is buggy!



Many projects fail



http://www.cs.ucc.ie/~herbert/CS4504/X%20Software%20Risks/laacm0512-Why-Big-SW-Projects-Fail0503Humphrey.pdf

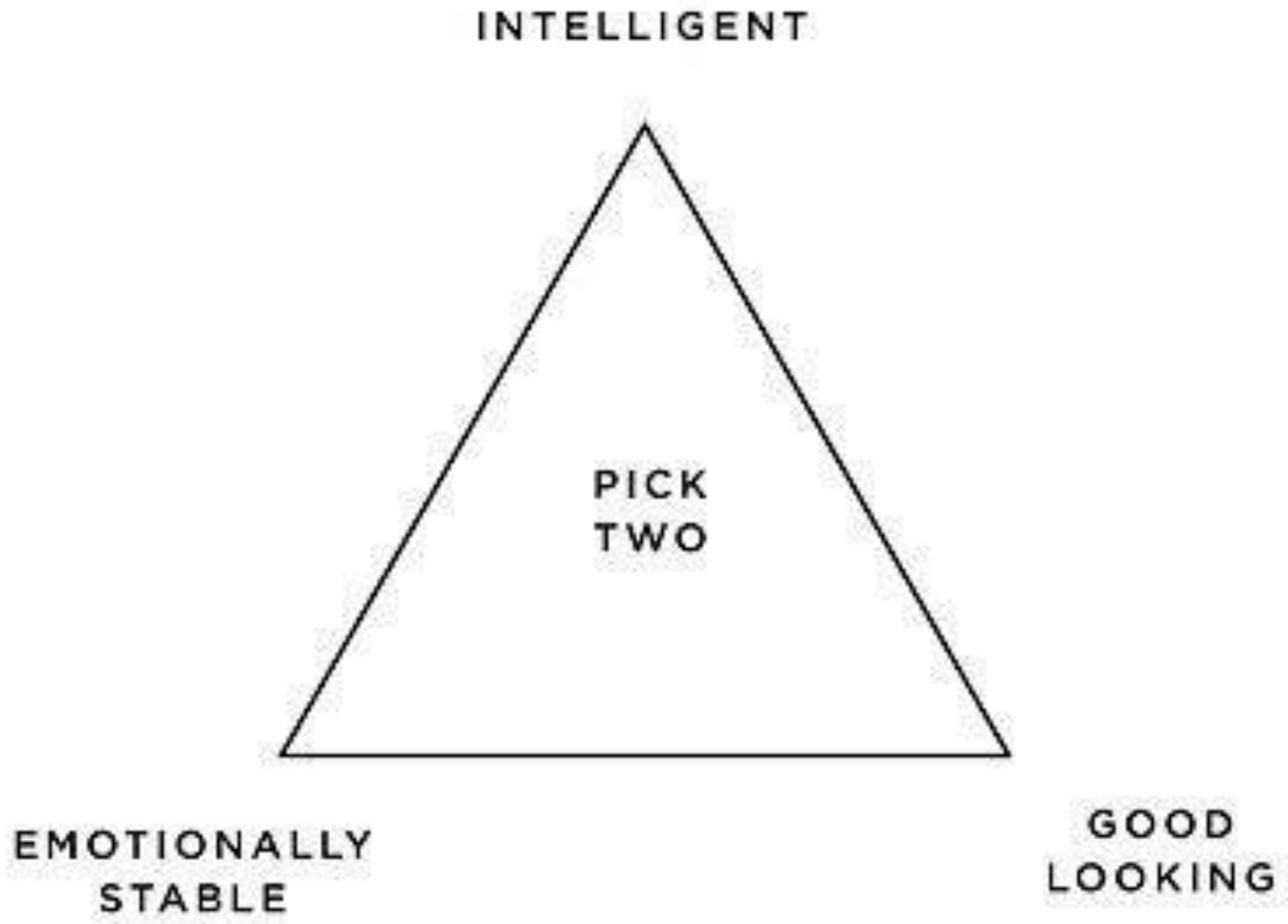


Hmm, I can do programming!

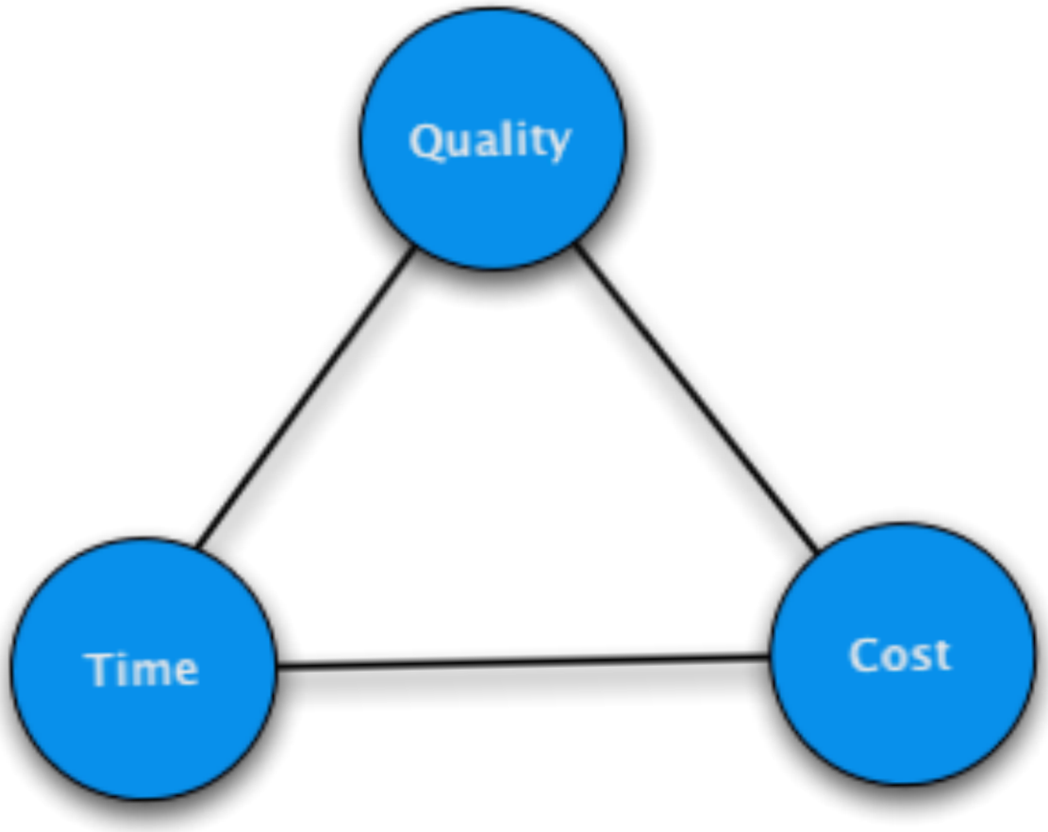
- I'm actually good at it, so no problem for me
- But you need collaborations
- Collaboration sounds easy and fun!
  - Why we fight? (wife&husband, girl&boy friends)
- Software is naturally complicated
  - Need to think of all cases
  - Need to think them in advances

Magical Triangle

Magical Triangle



Magical Triangle



http://www.traue.com/pm/page4.html

What is Software Engineering?

- The application of
  - systematic
  - disciplined
  - quantifiable approach
- to
  - design
  - development
  - operation
  - maintenance

http://en.wikipedia.org/wiki/Software_engineering

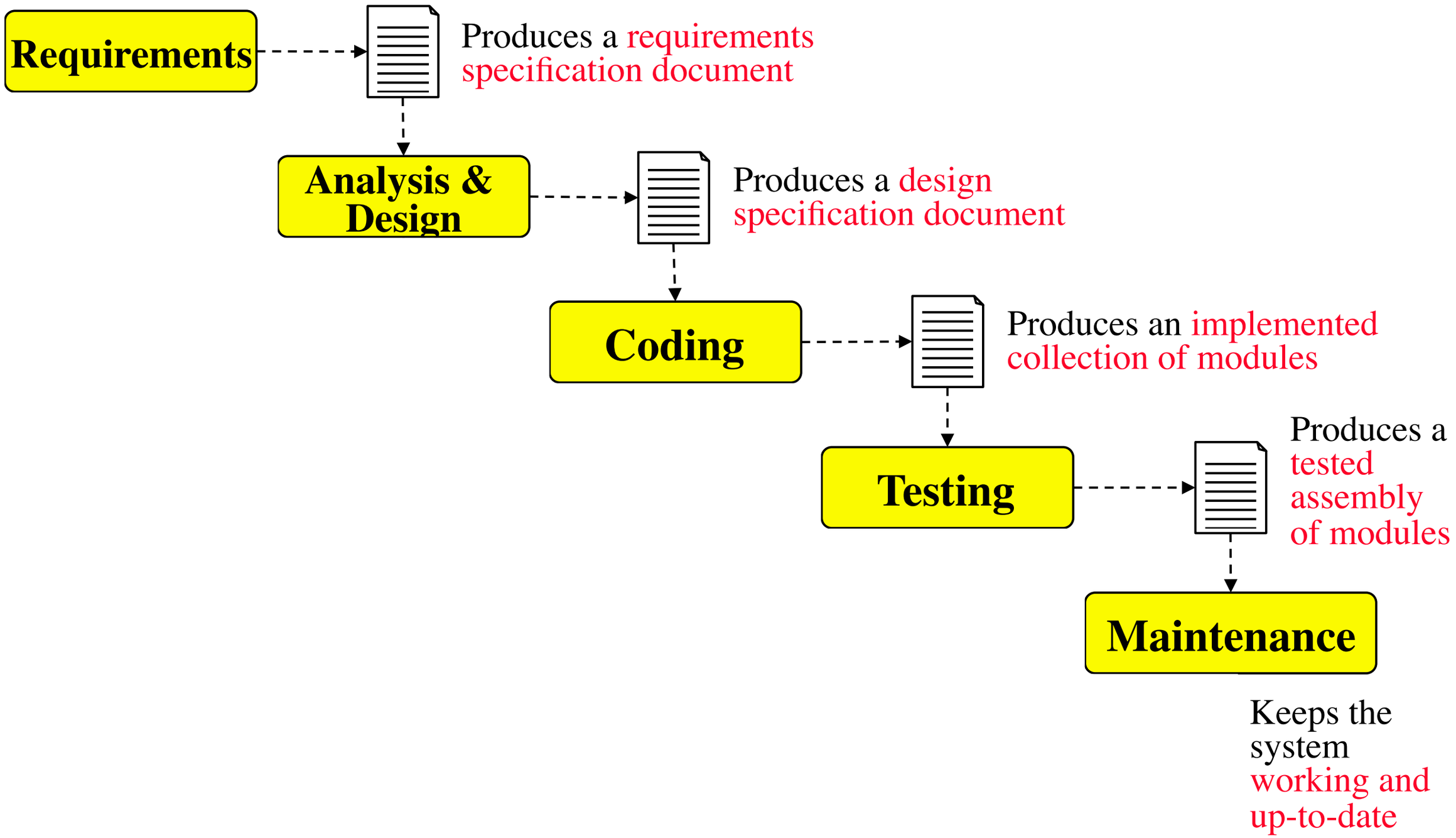
SE Methodologies

- Requirements/Documentation: trello
- Version control system: git
- Issue tracking system: github
- Code review/pair programming: github
- Testing: JUnit
- Continus Integration: Jenkins, Travis

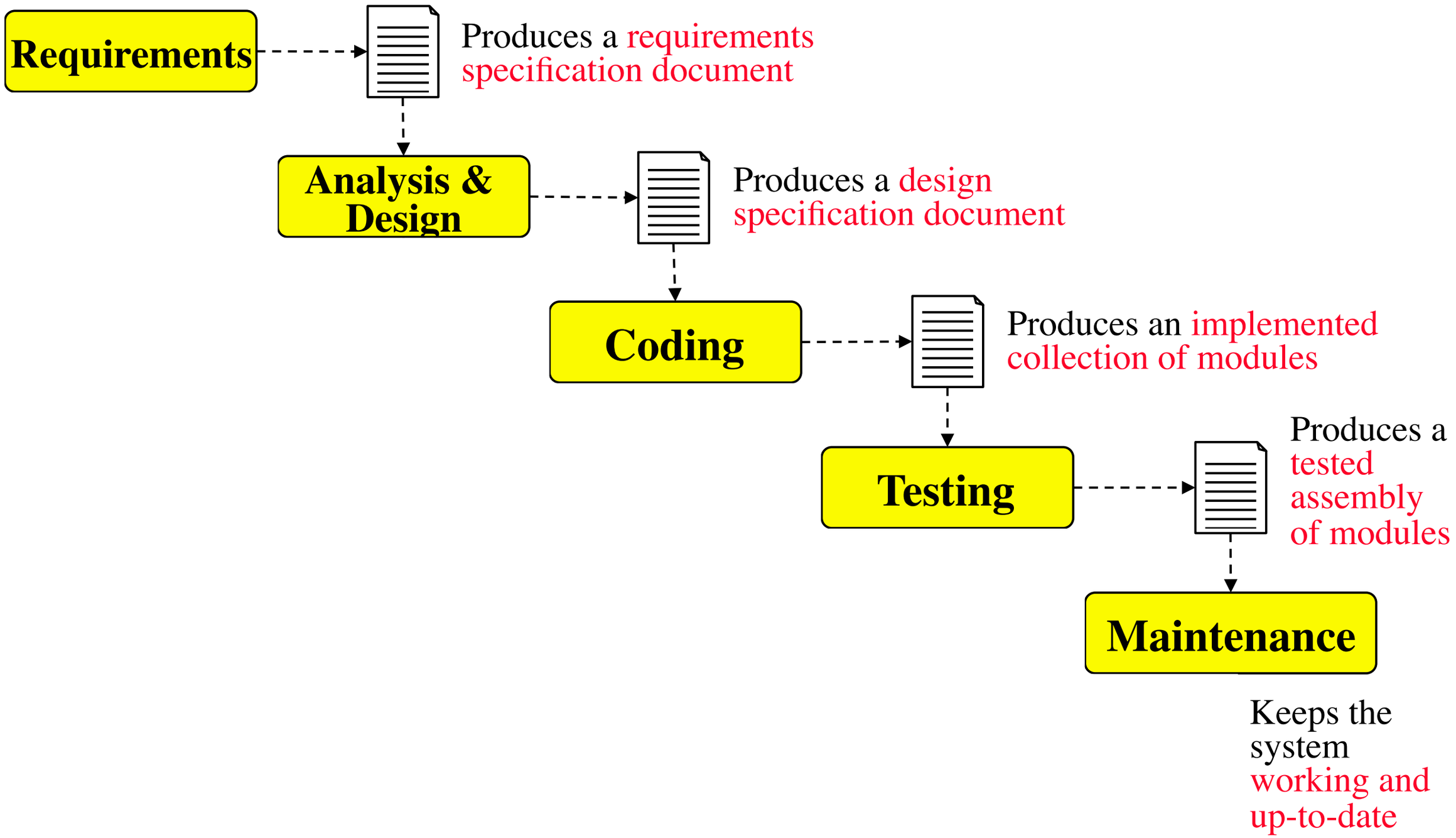
SE process/requirements

SE process/requirements

WATERFALL SOFTWARE DEVELOPMENT PROCESS



WATERFALL SOFTWARE DEVELOPMENT PROCESS



WHAT IS A REQUIREMENT?

A *requirement* is a feature that the system must have or a constraint that it must satisfy to be accepted by the client.

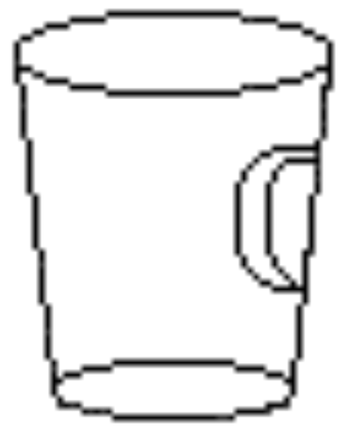
- "Something required; something wanted or needed." *Webster's Ninth New Collegiate Dictionary*

- "a condition or capability need by a user to solve a problem or achieve an objective." *IEEE Standard 729*

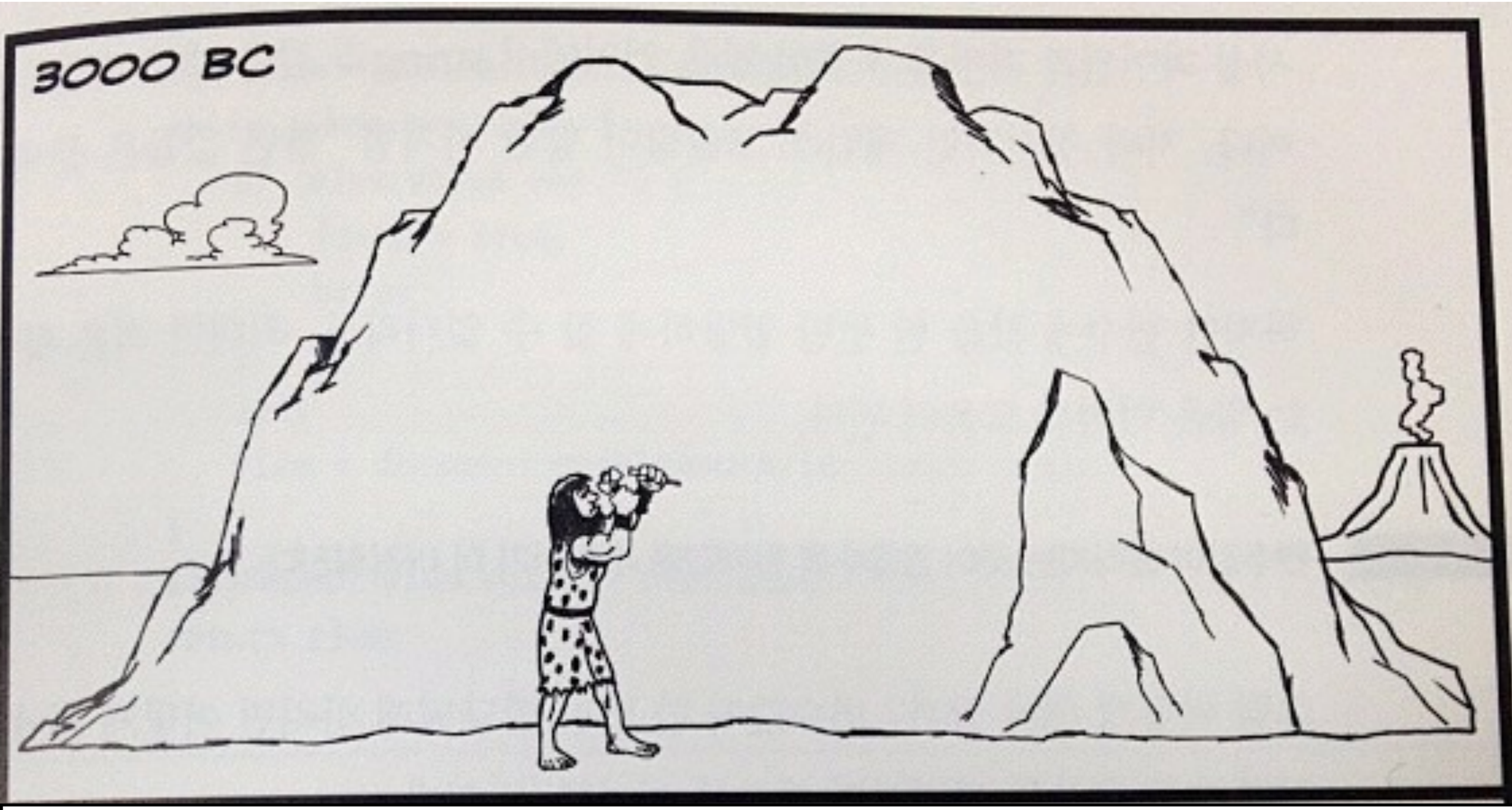
- A complete statement of **what** the system will do *without* referring to **how** it will do it.

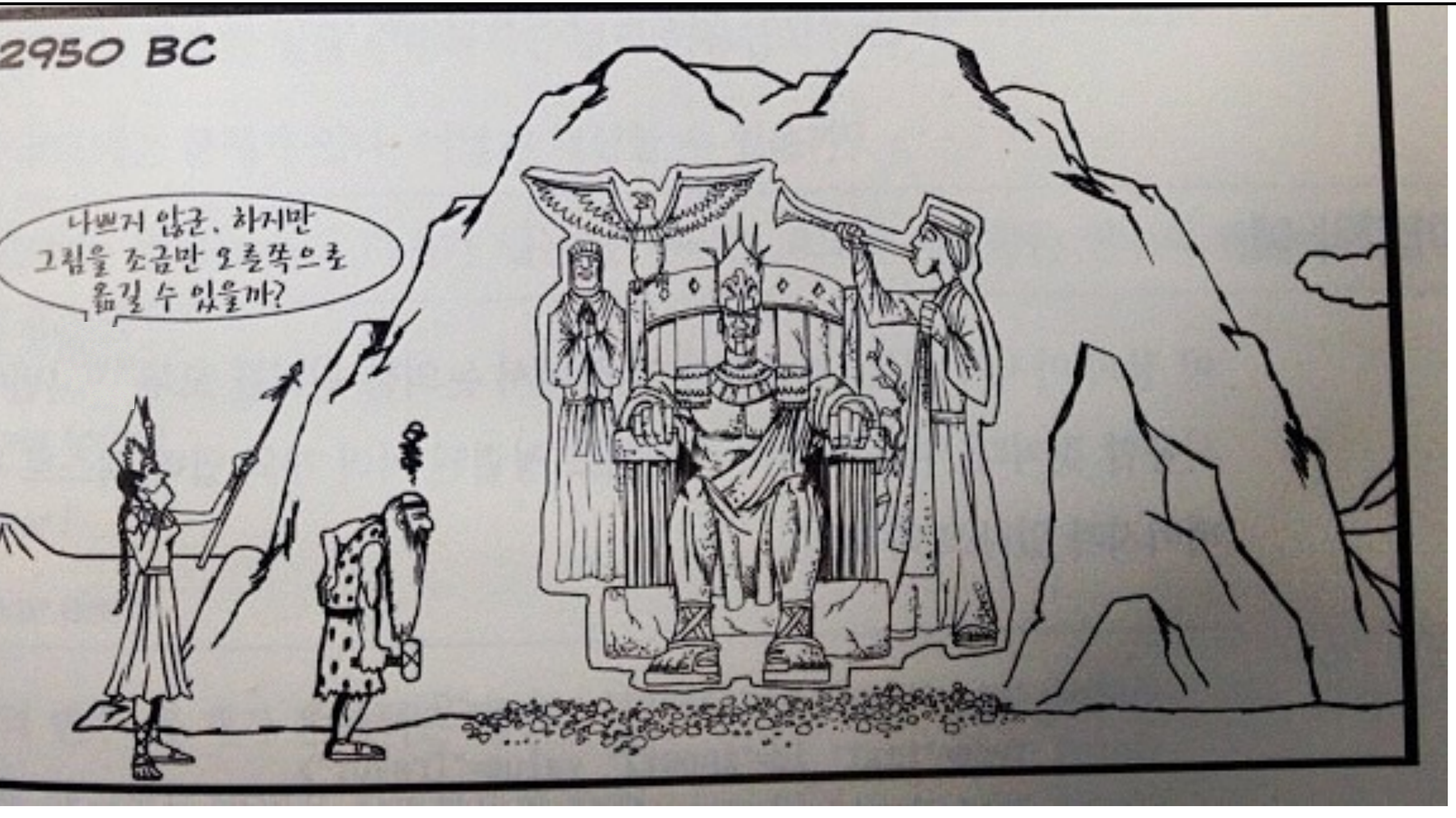
Hot drink

- Needed the capability to drink hot liquids
- Requirements
  - The thing shall contain up to 8 ounces of hot liquid.
  - The thing shall have a handle.
  - No details how to make the thing!



http://www.jiludwig.com/What_vs_How.html


3000 BC


2950 BC
나쁘지 않군. 하지만
그림을 조금만 오른쪽으로
옮길 수 있을까?

THE IMPORTANCE OF REQUIREMENTS CAPTURE

Why do we need to capture and document requirements?

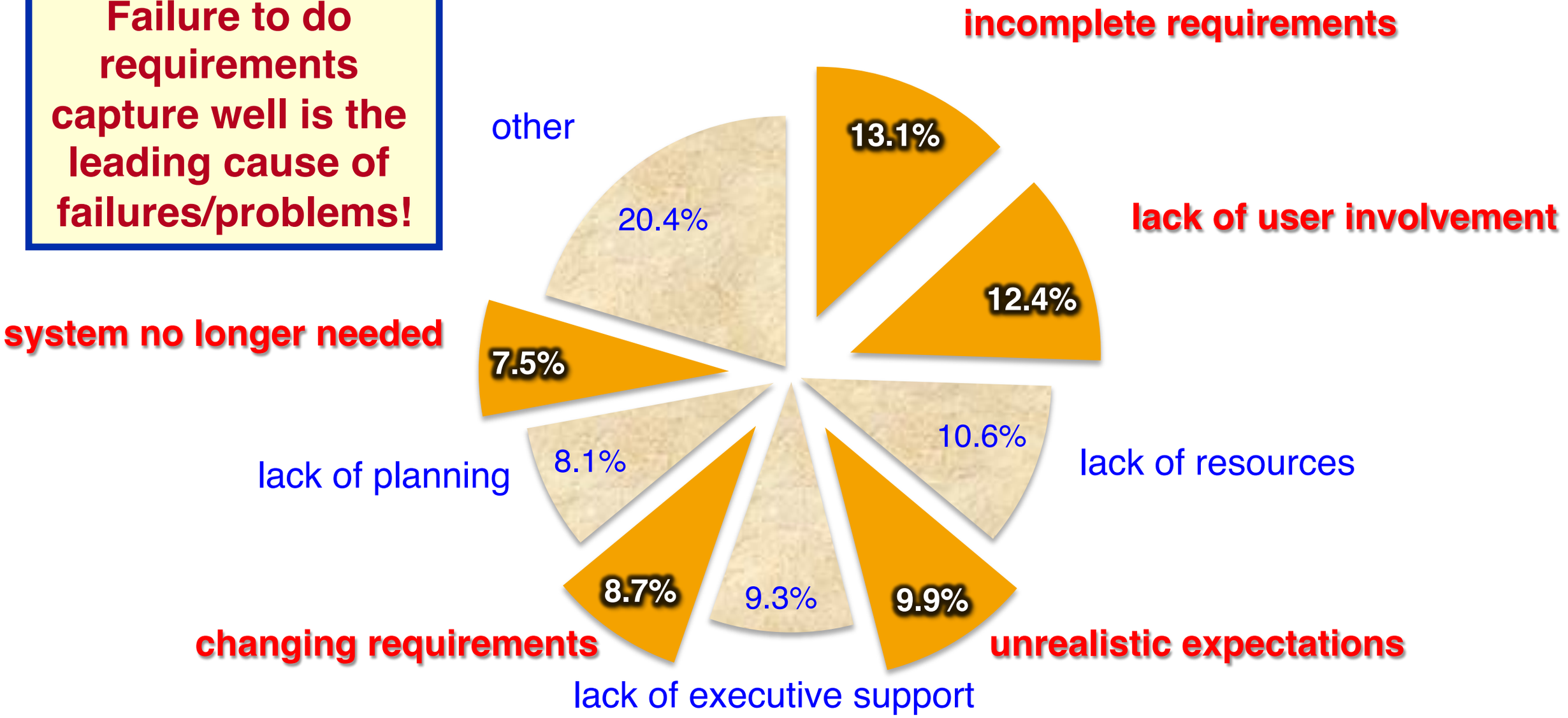
- Our brain is optimistic!
  - Think of a night dream.
  - You think you remember everything.
  - But when you try to remember details, you can't!

- The same thing happens in software development.
  - You think you know every detail.
  - But, during coding you find many things are undefined or ambiguous.

THE IMPORTANCE OF REQUIREMENTS CAPTURE

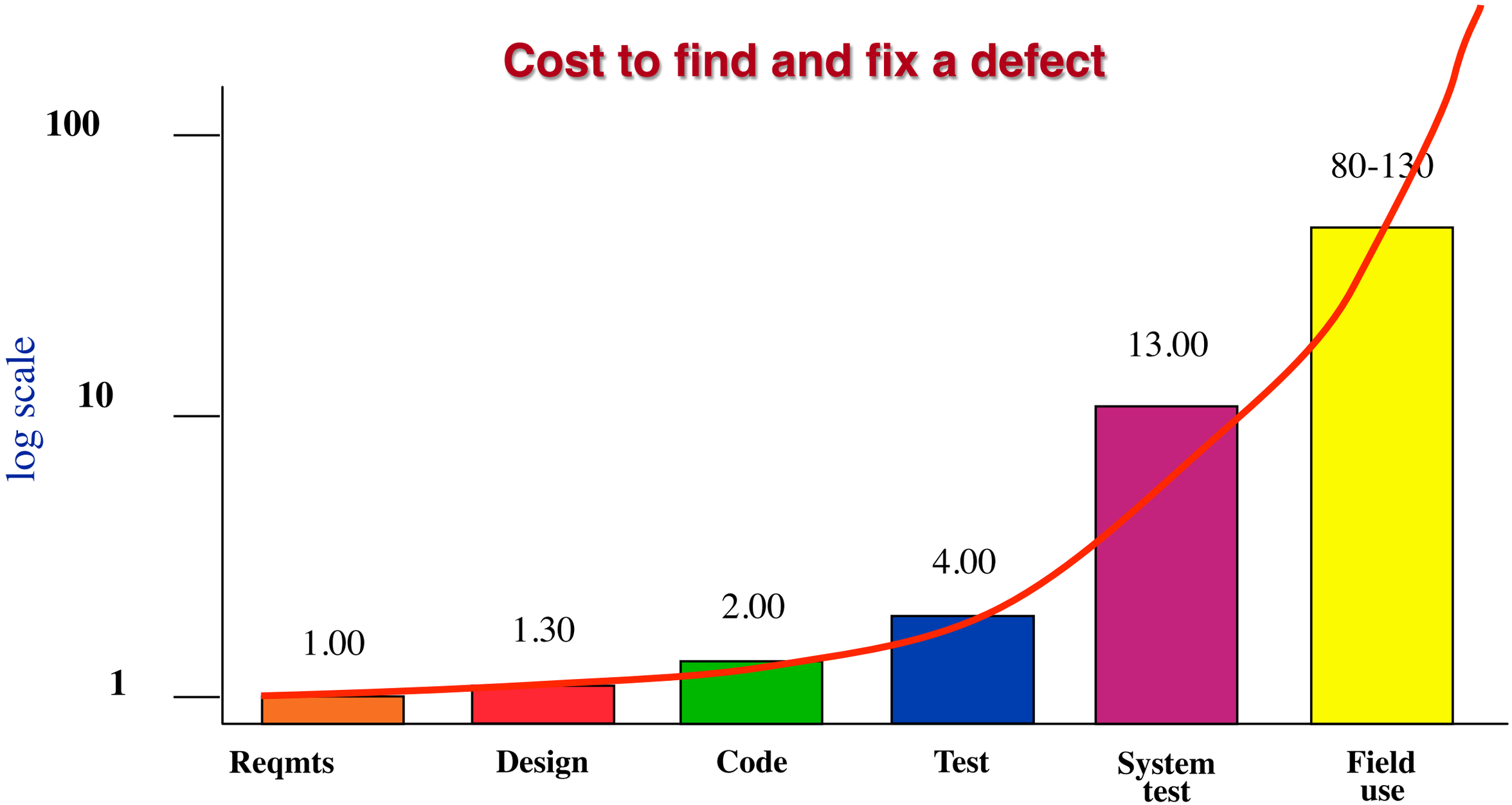
Reasons for failure of or problems with software development

Failure to do requirements capture well is the leading cause of failures/problems!



THE IMPORTANCE OF REQUIREMENTS CAPTURE

Cost to find and fix a defect



Good requirements capture helps reduce the cost of software development.

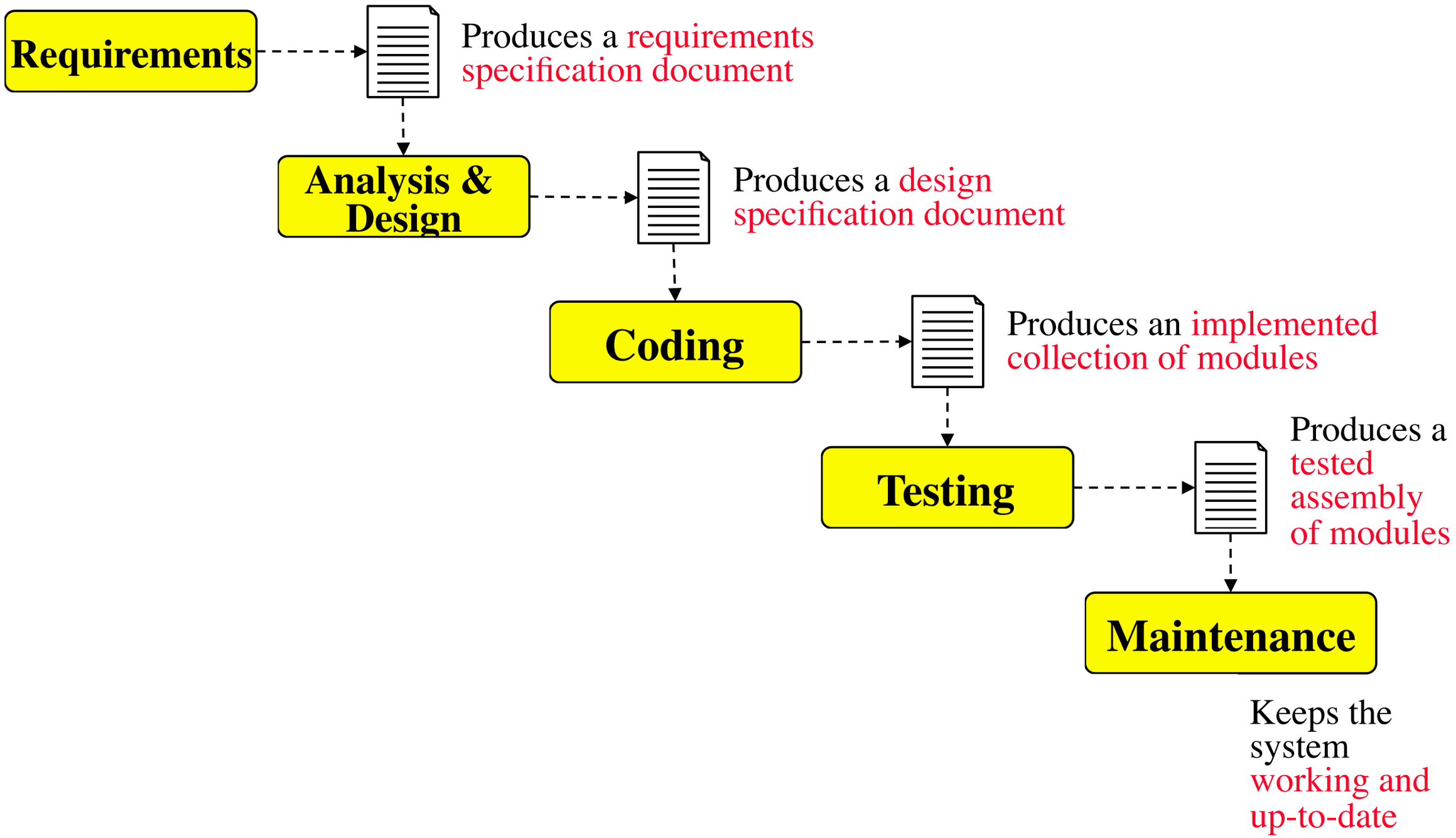
The structure of a requirements document 1

Chapter	Description
Preface	This should define the expected readership of the document and describe its version history, including a rationale for the creation of a new version and a summary of the changes made in each version.
Introduction	This should describe the need for the system. It should briefly describe the system's functions and explain how it will work with other systems. It should also describe how the system fits into the overall business or strategic objectives of the organization commissioning the software.
Glossary	This should define the technical terms used in the document. You should not make assumptions about the experience or expertise of the reader.
User requirements definition	Here, you describe the services provided for the user. The nonfunctional system requirements should also be described in this section. This description may use natural language, diagrams, or other notations that are understandable to customers. Product and process standards that must be followed should be specified.
System architecture	This chapter should present a high-level overview of the anticipated system architecture, showing the distribution of functions across system modules. Architectural components that are reused should be highlighted.

The structure of a requirements document II

Chapter	Description
System requirements specification	This should describe the functional and nonfunctional requirements in more detail. If necessary, further detail may also be added to the nonfunctional requirements. Interfaces to other systems may be defined.
System models	This might include graphical system models showing the relationships between the system components and the system and its environment. Examples of possible models are object models, data-flow models, or semantic data models.
System evolution	This should describe the fundamental assumptions on which the system is based, and any anticipated changes due to hardware evolution, changing user needs, and so on. This section is useful for system designers as it may help them avoid design decisions that would constrain likely future changes to the system.
Appendices	These should provide detailed, specific information that is related to the application being developed; for example, hardware and database descriptions. Hardware requirements define the minimal and optimal configurations for the system. Database requirements define the logical organization of the data used by the system and the relationships between data.
Index	Several indexes to the document may be included. As well as a normal alphabetic index, there may be an index of diagrams, an index of functions, and so on.

WATERFALL SOFTWARE DEVELOPMENT PROCESS



Manifesto for Agile Software Development

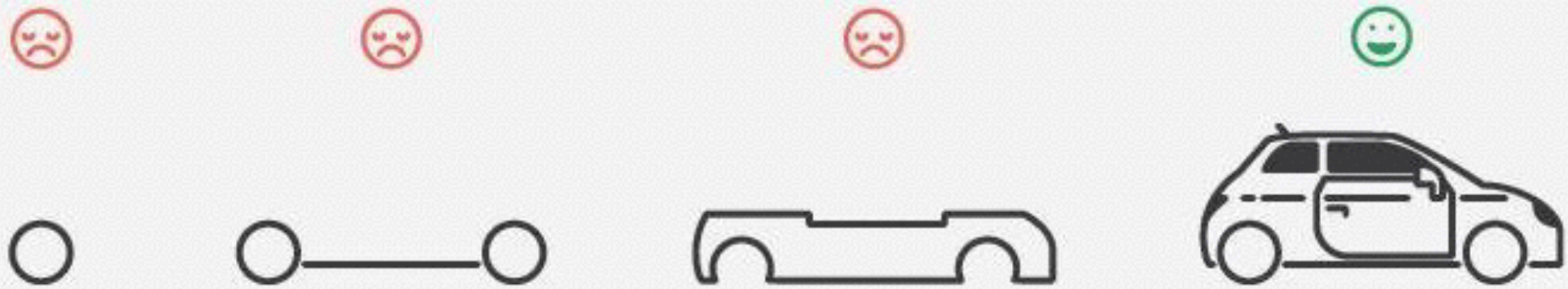
We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

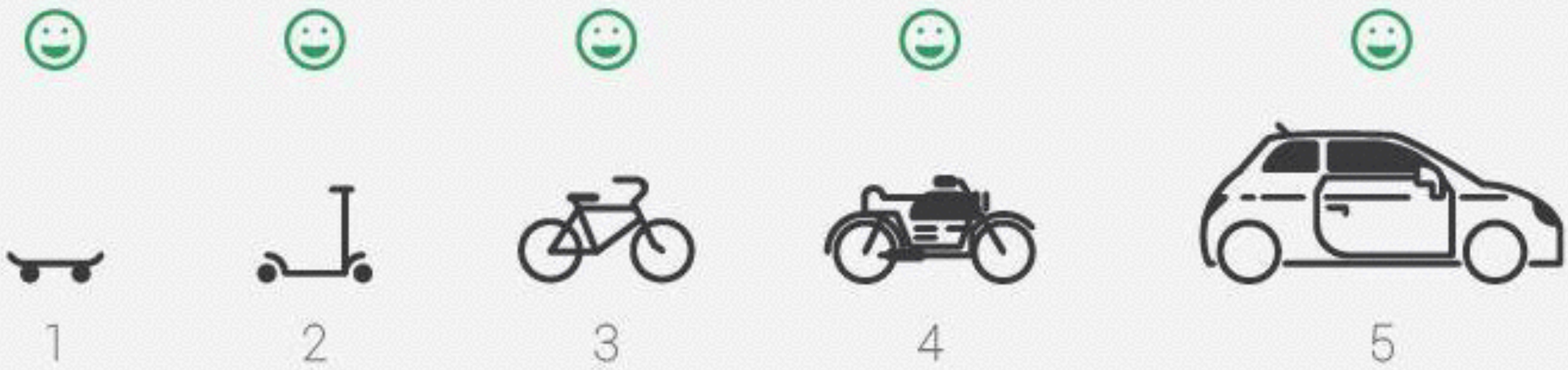
That is, while there is value in the items on
the right, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

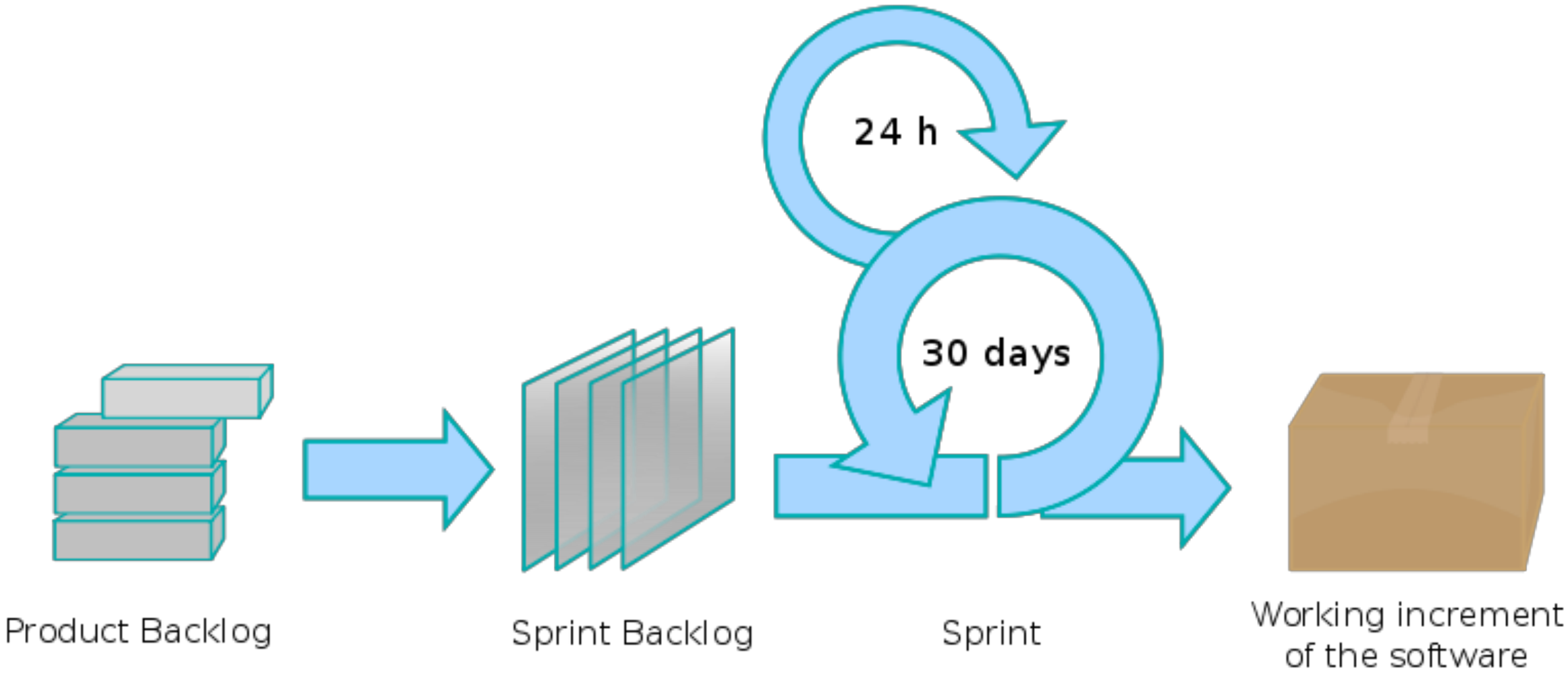
How not to build a minimum viable product



How to build a minimum viable product



SCRUM



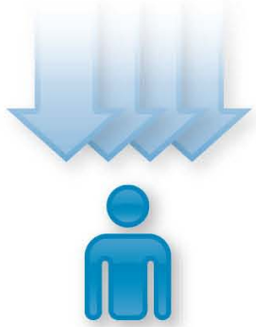
http://en.wikipedia.org/wiki/Scrum_%28development%29

SCRUM Tutorial

- http://www.youtube.com/watch?v=Q5k7a9YEoUI (Old)
- http://www.youtube.com/watch?v=XU0llRItyFM (New)
- http://www.youtube.com/watch?v=D8vT7G0WATM&list=PLF6BFA8BAEDF6CE70 (12 videos)

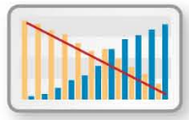
The Agile: Scrum Framework at a glance

Inputs from Executives, Team, Stakeholders, Customers, Users



Product Owner

The Team



Burndown/up Charts

Scrum Master

Daily Scrum Meeting

Every 24 Hours

1-4 Week Sprint

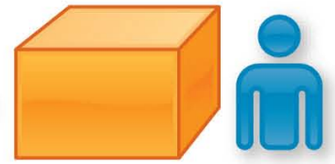
Sprint Review

Product Backlog

Sprint Planning Meeting

Sprint Backlog

Sprint end date and team deliverable do not change



Finished Work

Sprint Retrospective

neon rain interactive

AGILE FOR ALL

User Story

- Ron Jeffries offers a simple yet effective way to think about user stories (Jeffries 2001).
- He describes them as the three Cs
  - Card
  - Conversation
  - Confirmation

Rubin, Kenneth S. Essential Scrum: A Practical Guide to the Most Popular Agile Process (Addison-Wesley Signature Series (Cohn))

User Story - Cards

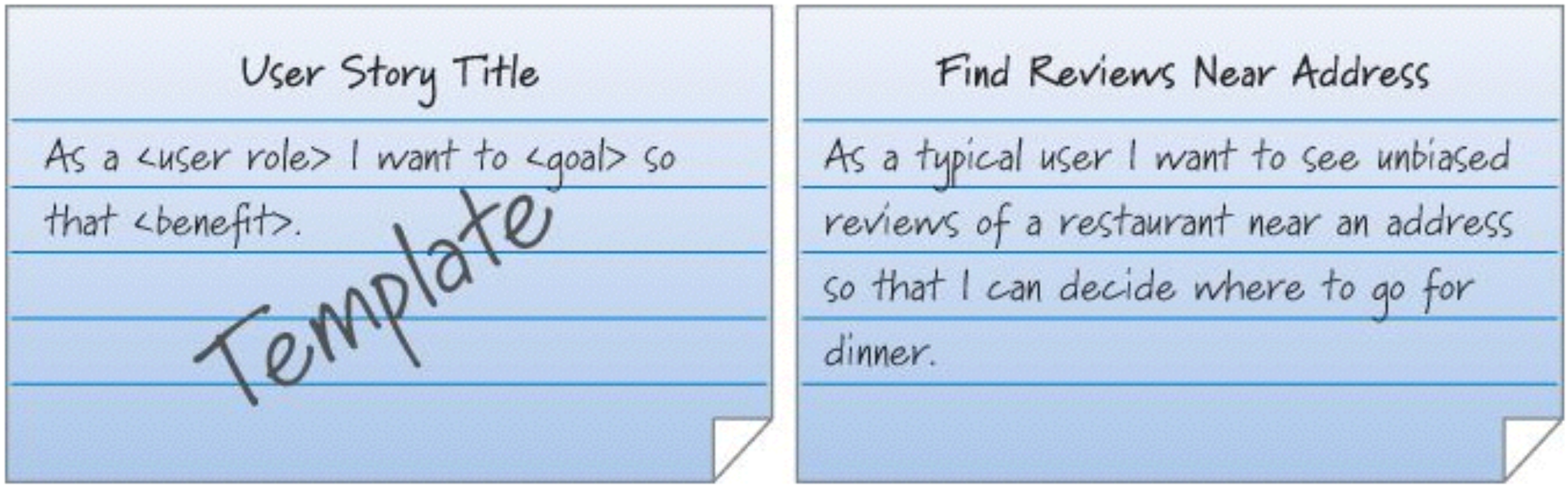


Figure 5.2. A user story template and card

Rubin, Kenneth S. Essential Scrum: A Practical Guide to the Most Popular Agile Process (Addison-Wesley Signature Series (Cohn))

User Story - Conversation

- The details of a requirement are *exposed and communicated* in a conversation among the development team, product owner, and stakeholders.
- The user story is simply *a promise (starting point)* to have that conversation.

Rubin, Kenneth S. Essential Scrum: A Practical Guide to the Most Popular Agile Process (Addison-Wesley Signature Series (Cohn))

User Story - Conversation

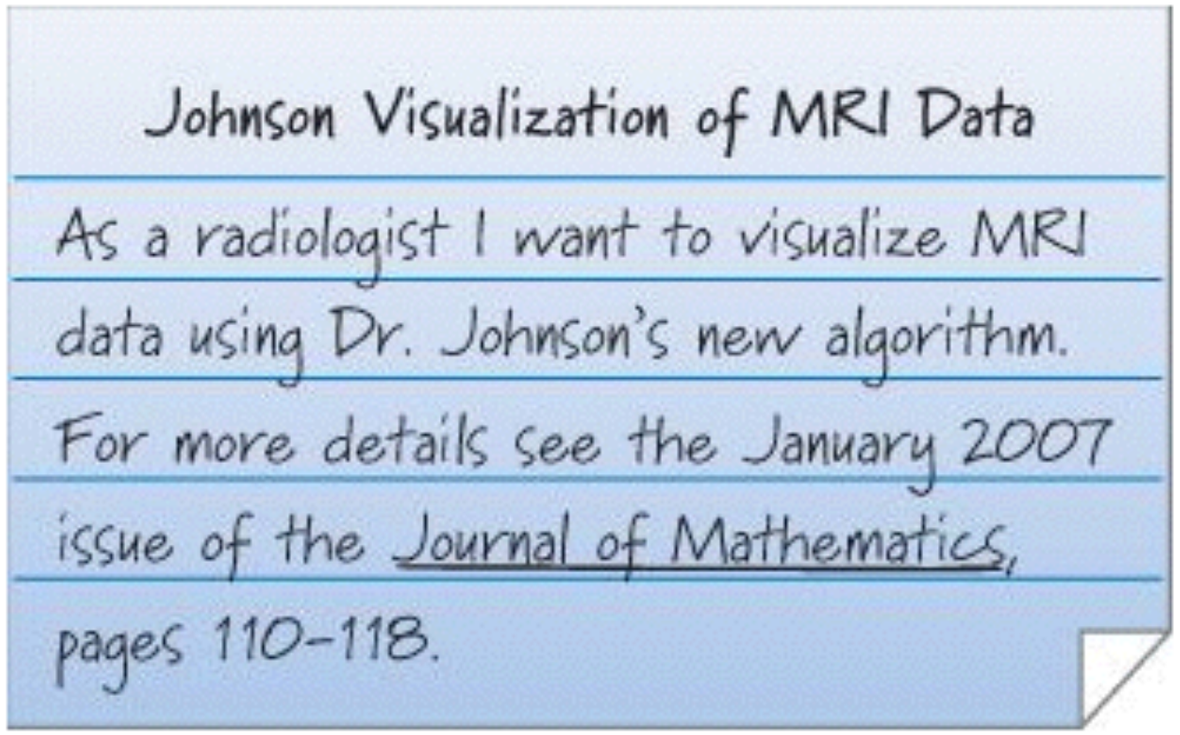


Figure 5.3. User story with additional data attached

Rubin, Kenneth S. Essential Scrum: A Practical Guide to the Most Popular Agile Process (Addison-Wesley Signature Series (Cohn))

User Story - Confirmation

- A user story also contains *confirmation information* in the form of conditions of satisfaction.
- These are *acceptance criteria* that clarify the desired behavior.
- They are used by the development team to *better understand what to build* and *test* to confirm that the user story has been implemented to their satisfaction.

Rubin, Kenneth S. Essential Scrum: A Practical Guide to the Most Popular Agile Process (Addison-Wesley Signature Series (Cohn))

User Story - Confirmation

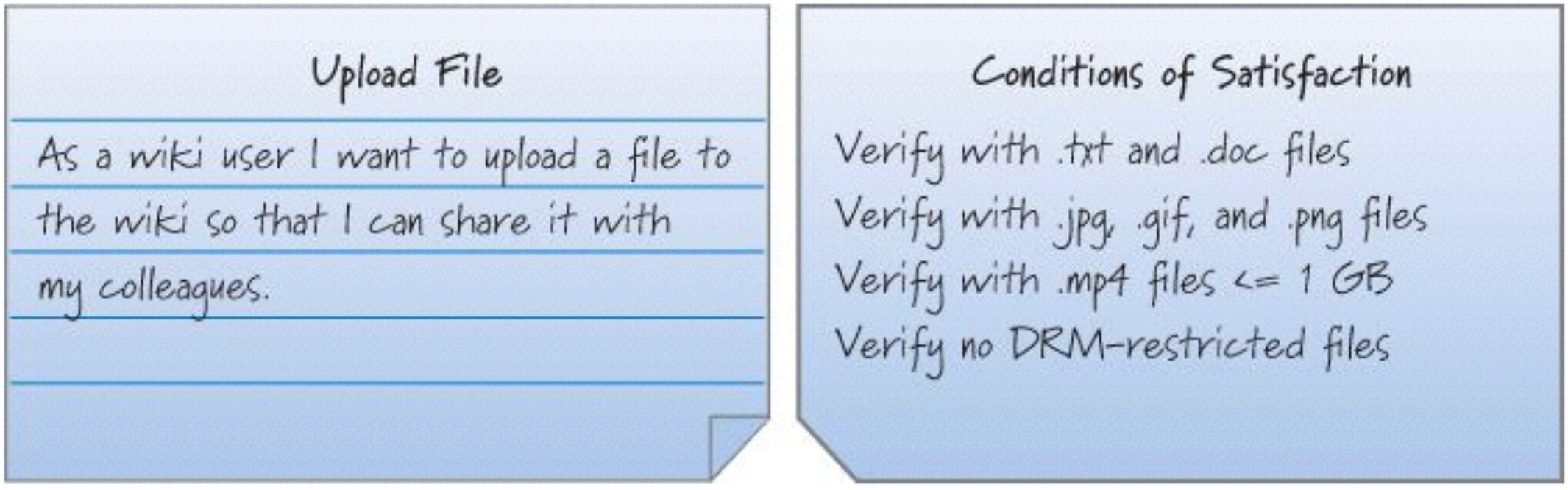


Figure 5.4. User story conditions of satisfaction

INVEST in good stories

- Independent
- Negotiable
- Valuable
- Estimable
- Small (Size appropriately)
- Testable

Story map

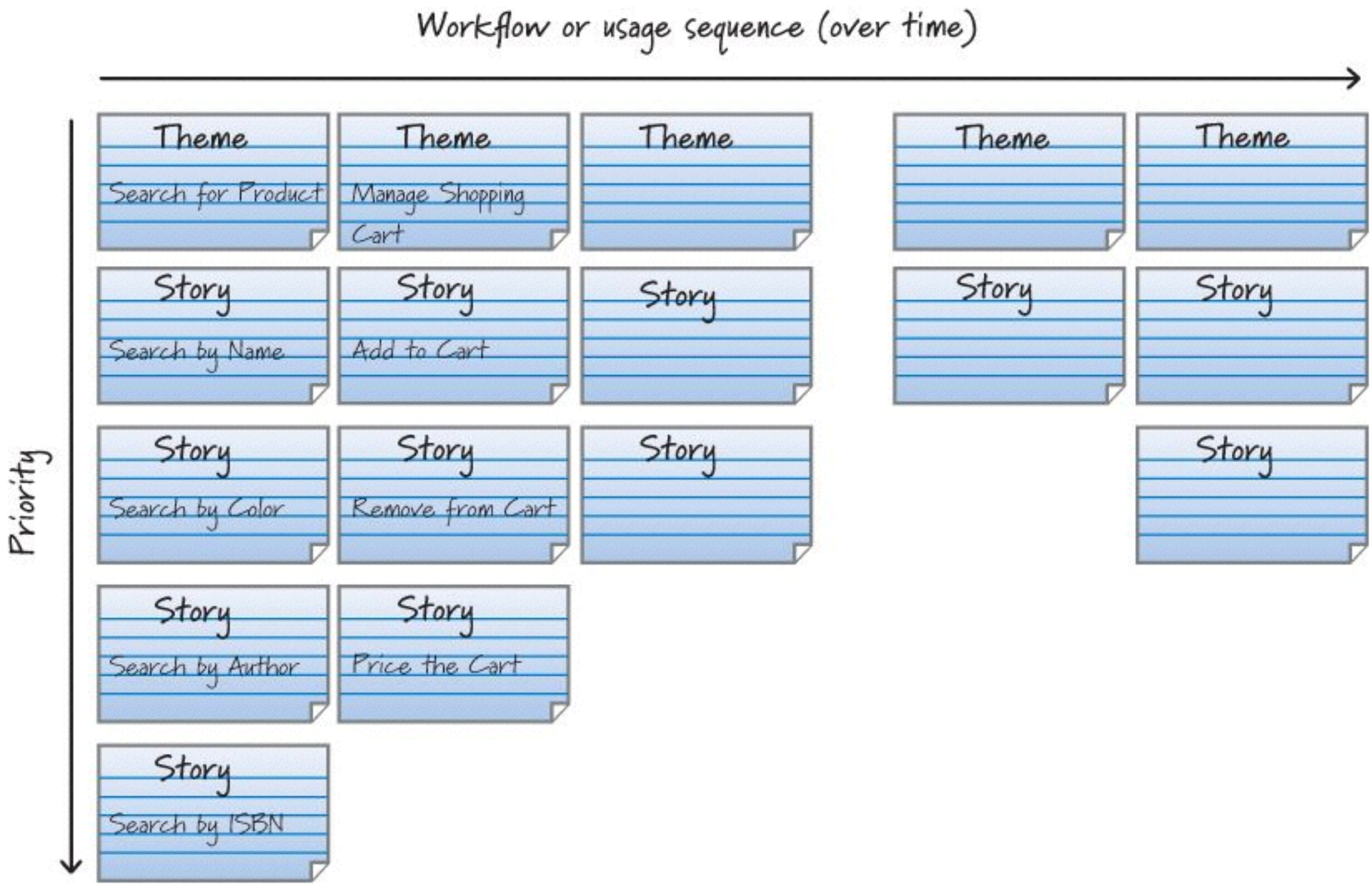


Figure 5.13. Story map

https://theagilecoder.files.wordpress.com/2013/11/scrumfortrelloboard.png

Trello

Burndown for Trello - Product Backlog

Public

Backlog (noburn)

FOUND A BUG OR HAVE A SUGGESTION? PLEASE COMMENT ON THIS CARD! :)

2 votes 55

Only include the StripeJs (from their site) if the user is loggedout, on a free account, or on the account page. Paid users don't need it on the main app & slows down our DomReady

1 vote

Mobile skin

Add ability to ignore holidays, specific days, etc..

4 votes 2

Skin Revamp

Doing

Fix ScrumForTrello Firefox performance issue

Make screencast tutorials

1 vote 1 0/7

Make the system send me an email when I get a Mailbox message since that's being used as another avenue for support by users & it should be as snappy as emailing.

Quick Survey

1 vote 0/8

On Test Server

(8) Toggl extension prototype (entering estimates) [8.5]

3 votes 7/9

Done

Re-release the Safari version (the currently released version doesn't even have completed-points-picker).

(8) Finally solve relogin [3] GOT IT! :D

2 votes 10 6/7

(2) Figure out a way to prevent the constant re-authentication when showing the code in the browser extension.

1 vote 1

Create a messaging system so that we can contact free users (we only get email addresses during signup for the paid account). [3.5]

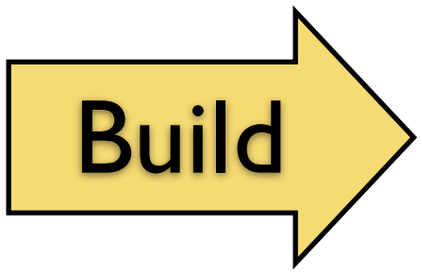
1

SE Methodologies

- Requirements/Documentation: trello
- Version control system: git
- Issue tracking system: github
- Code review/pair programming: github
- Testing: JUnit
- Continus Integration: Jenkins

Version Control System

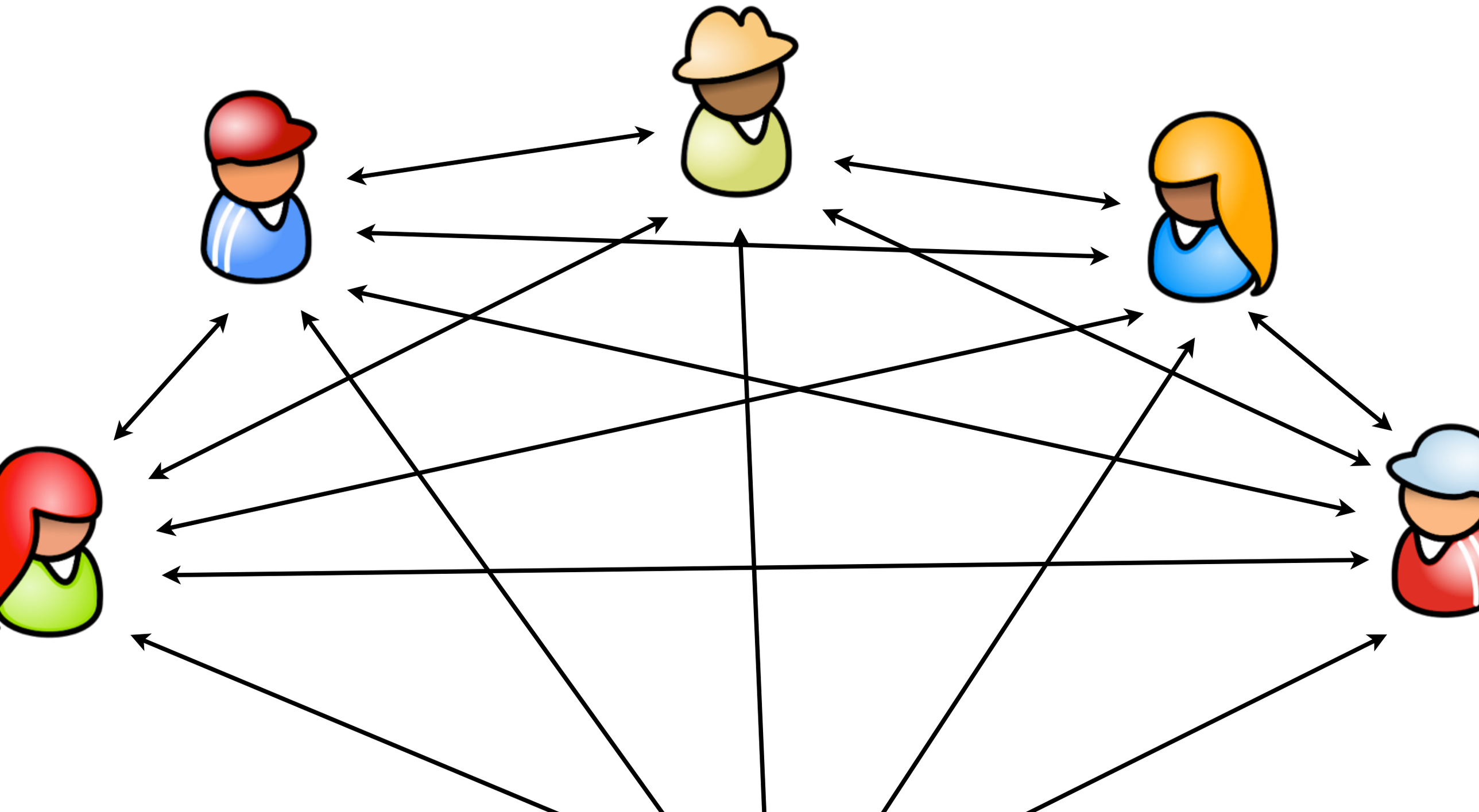
Software development



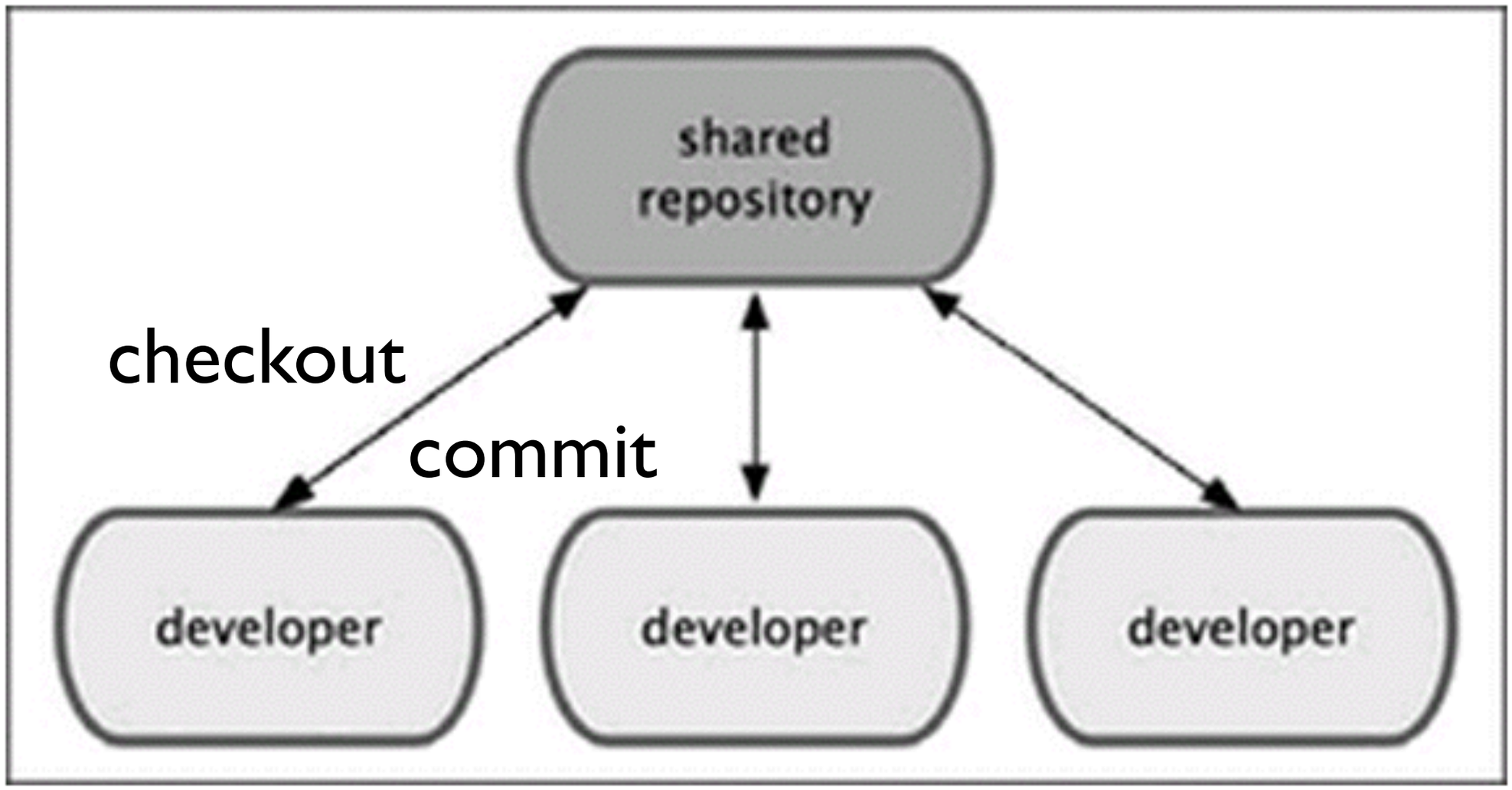
Software development

Build

Collaboration



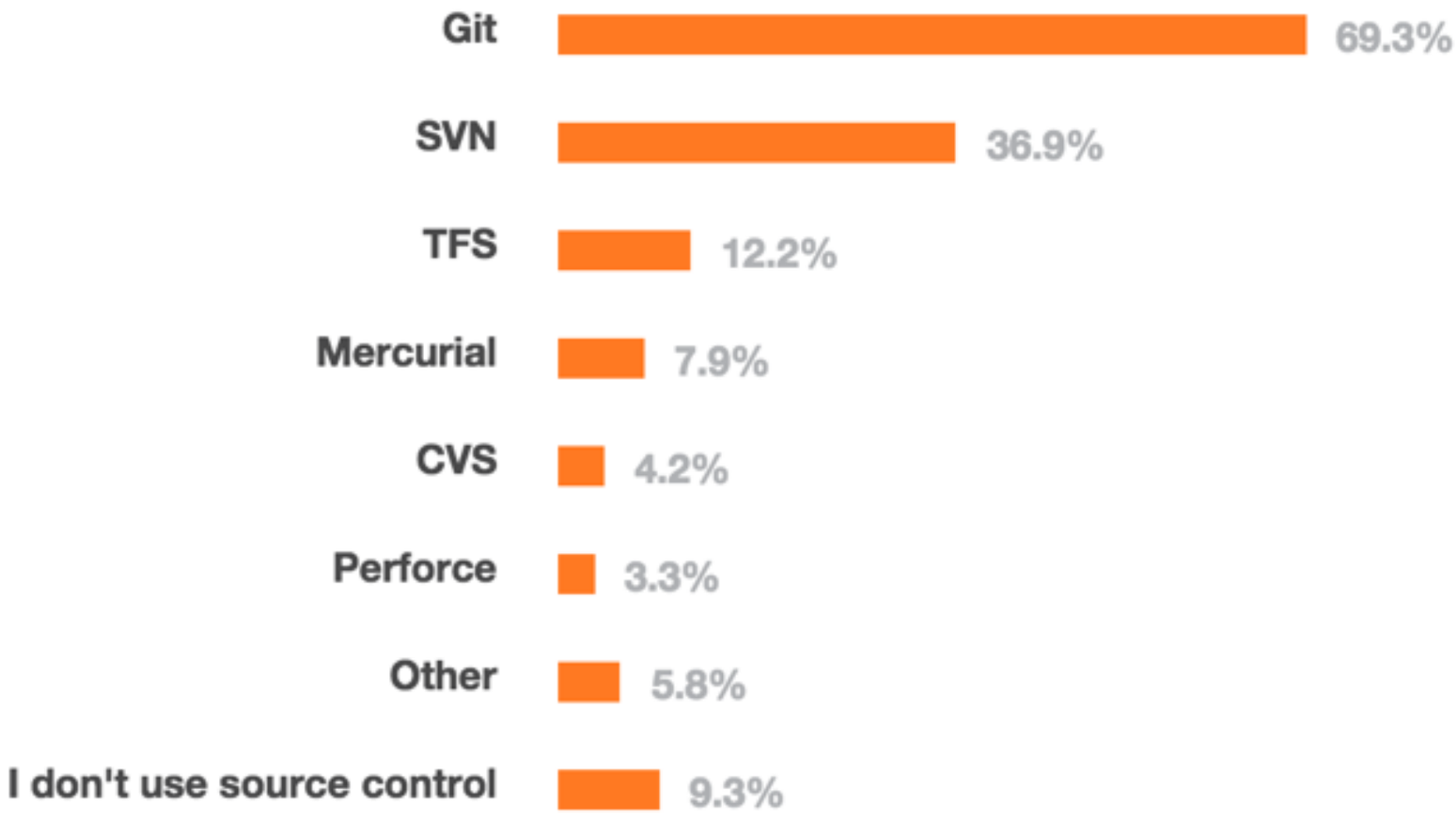
Version Control System



stackoverflow

2015 Developer Survey

VI. SOURCE CONTROL



16,694 responses

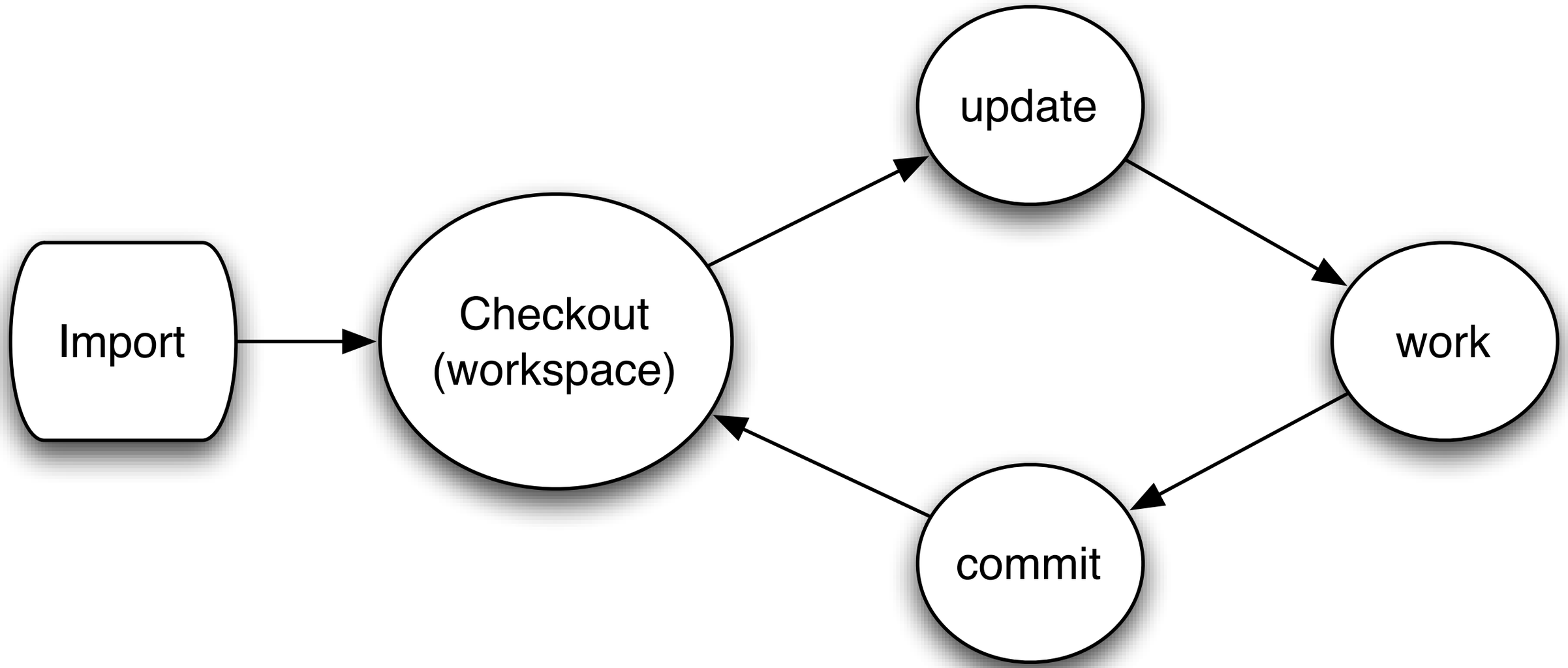
http://stackoverflow.com/research/developer-survey-2015#tech-sourcecontrol

Subversion (SVN)

- Open-source version control system
- Compatible successor to the widely used Concurrent Versions System (CVS)
- Change set based (rather than file revision based)
- Directory structure
  - trunk (usually have your projects in trunk)
  - tags
  - branches

http://en.wikipedia.org/wiki/Apache_Subversion

SVN - life cycle



SVN Workflow

- Create a repository (once)
- Import or checkout the repository
  - Work in your workspace

http://www.slideshare.net/intellibitz/subversion-user-guide

Typical work cycle

- Update your working copy
  - svn update
- Make changes
  - svn add
  - svn delete
  - svn copy
  - svn move
- Examine your changes
  - svn status
  - **svn diff**
  - svn revert
- Merge others' changes
  - svn merge
  - svn resolved
- Commit your changes
  - svn commit

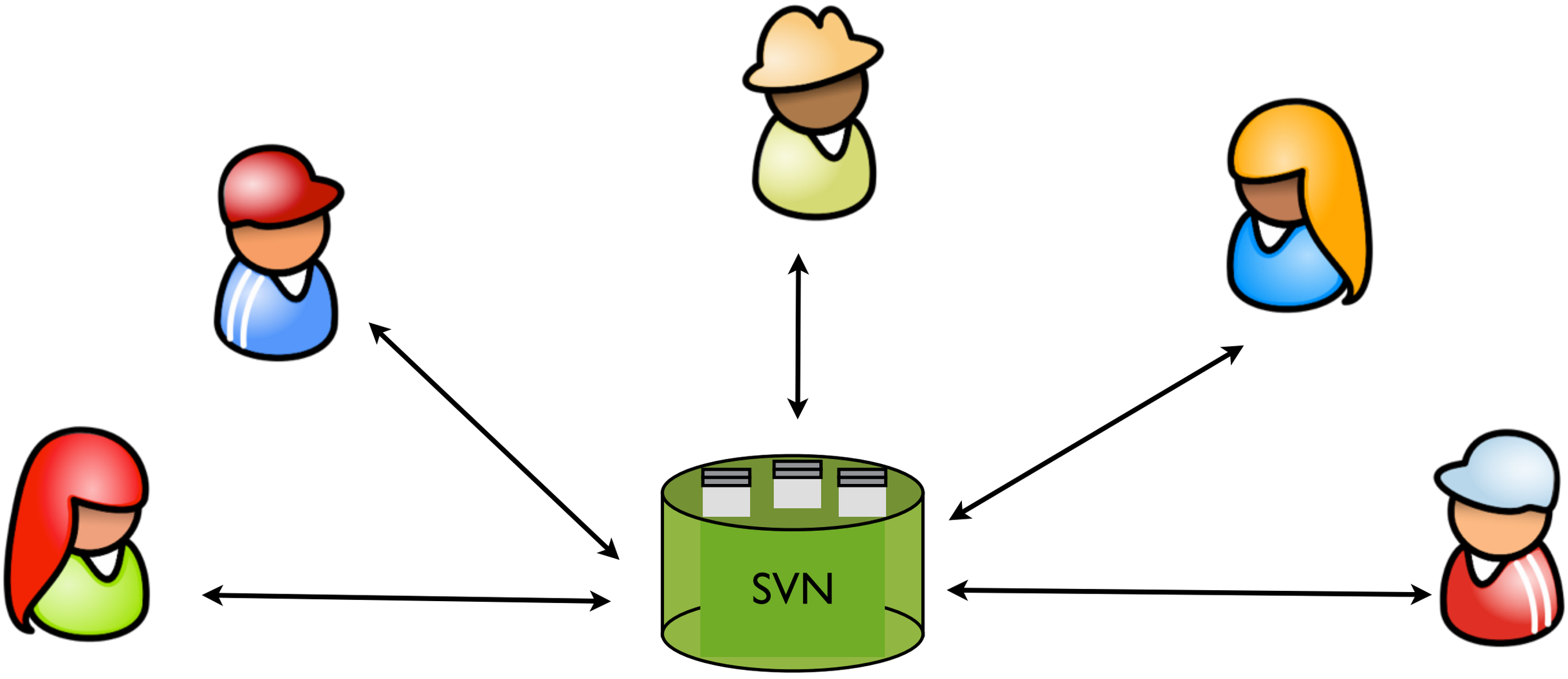
http://svnbook.red-bean.com/en/1.0/ch03s05.html

svn diff is handy

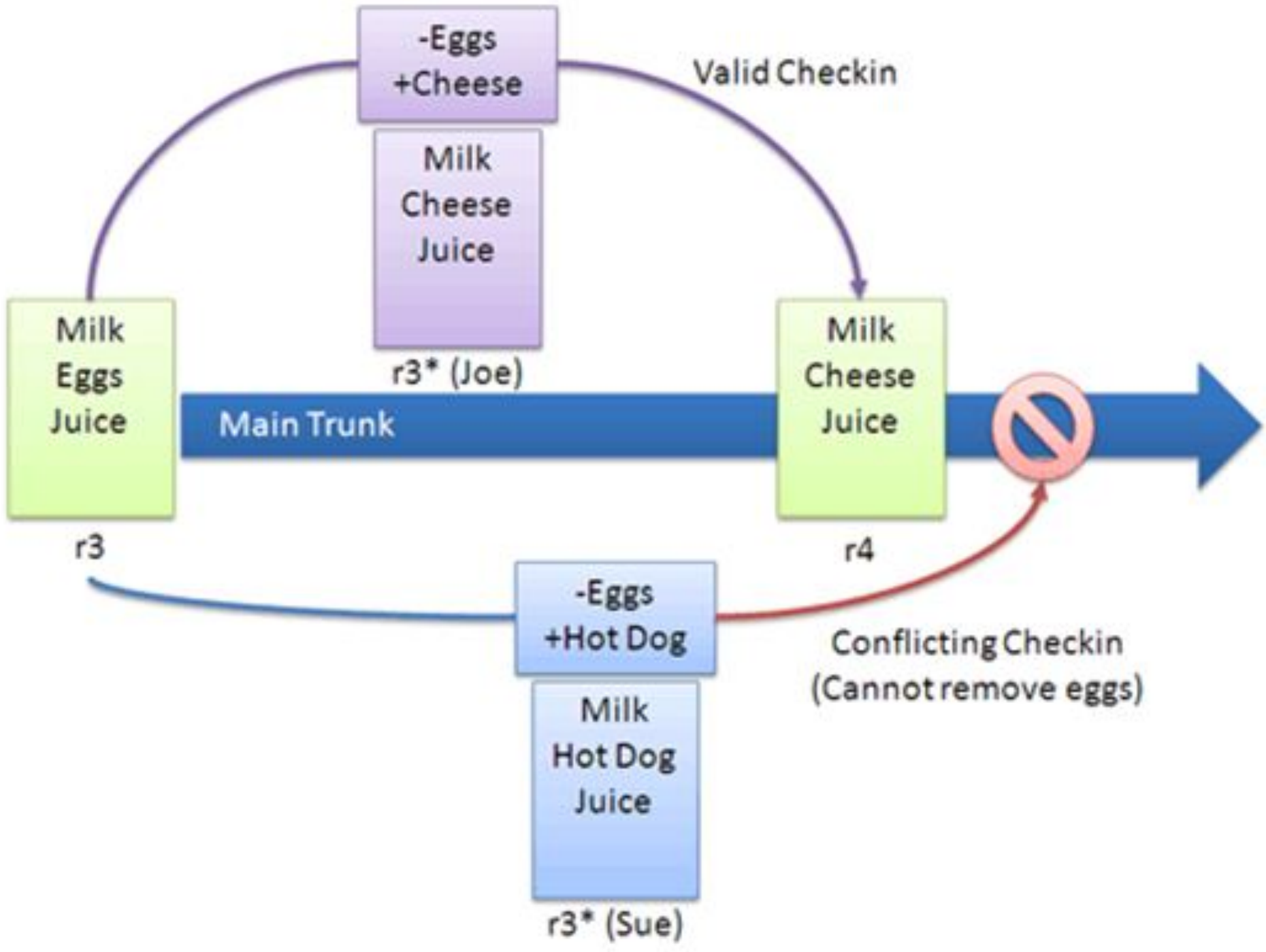
```
$ svn status
M       foo.c
$ svn diff

Index: foo.c
===================================================================
--- foo.c (revision 128)
+++ foo.c (working copy)
@@ -1003,7 +1003,7 @@
     return ERROR_ON_THE_G_STRING;

   /* Do something in a loop. */
-  for (i = 0; i < txns->nelts; i++)
+  for (i = 0; i < txns->nelts; i--)
     {
       status = do_something(i);
       if (status)
```

http://svnbook.red-bean.com/en/1.7/svn.ref.svn.c.patch.html

Now we can work simultaneously!



Conflicts



http://www.makebetterthings.com/wp-content/uploads/2010/07/basic_conflict_vcs_svn.jpg

Resolving conflicts

```
$ svn commit -m "Updated README"
Sending    README
Transmitting file data .svn: Commit failed (details follow):
svn: Out of date: '/myproject/README'
```

```
$ svn update
C   README
Updated to revision 6.
```

https://ariejan.net/2007/07/04/how-to-resolve-subversion-conflicts

Resolving conflicts

```
$cat README

<<<<<<< .mine
This is fun stuff!
=======
This is a documentation file
>>>>>>> .r6
```

https://ariejan.net/2007/07/04/how-to-resolve-subversion-conflicts

Resolving conflicts: What Should I do?

1. Scrap your changes, and go with the current work from your colleague.
2. Keep your changes, and dump whatever your colleague did.
3. Merge both versions to a new version

https://ariejan.net/2007/07/04/how-to-resolve-subversion-conflicts

1. Scrap your changes

```
$ svn revert README
Reverted 'README'
$ svn update README
At revision 6.
```

https://ariejan.net/2007/07/04/how-to-resolve-subversion-conflicts

2. Keep your changes, and dump whatever your colleague did.

https://ariejan.net/2007/07/04/how-to-resolve-subversion-conflicts

2. Keep your changes, and dump whatever your colleague did.

https://ariejan.net/2007/07/04/how-to-resolve-subversion-conflicts

3. Merge both versions to a new version

- Manually edit the file (README)
  - remove the marks and add what you want to do
- SVN won't commit this yet, so you need to make it is resolved!

```
$ svn resolved README
Resolved conflicted state of 'README'
```

https://ariejan.net/2007/07/04/how-to-resolve-subversion-conflicts

Avoiding conflicts

- It won't happen too often
  - SVN automatically merges if different parts are changed. (good or bad)?
- do svn update frequently
- svn lock/unclock
- Refactoring
  - better modularization

Git

git - the simple guide

just a simple guide for getting started with git. no deep shit ;)

Tweet 4,747

by Roger Dudler

credits to @tfnico, @fhd and Namics

this guide in deutsch, español, français, indonesian, italiano, nederlands, polski, português, русский, türkçe, မြန်မာ, 日本語, 中文, 한국어 Vietnamese

please report issues on github

http://rogerdudler.github.io/git-guide/

create a new repository

create a new directory, open it and perform a

```
git init
```

to create a new git repository.

http://rogerdudler.github.io/git-guide/

workflow

your local repository consists of three "trees" maintained by git. the first one is your `Working Directory` which holds the actual files. the second one is the `Index` which acts as a staging area and finally the `HEAD` which points to the last commit you've made.

http://rogerdudler.github.io/git-guide/

add & commit

You can propose changes (add it to the **Index**) using

```
git add <filename>
```

```
git add *
```

This is the first step in the basic git workflow. To actually commit these changes use

```
git commit -m "Commit message"
```

Now the file is committed to the **HEAD**, but not in your remote repository yet.

http://rogerdudler.github.io/git-guide/

branching

Branches are used to develop features isolated from each other. The *master* branch is the "default" branch when you create a repository. Use other branches for development and merge them back to the master branch upon completion.

branching

create a new branch named "feature_x" and switch to it using

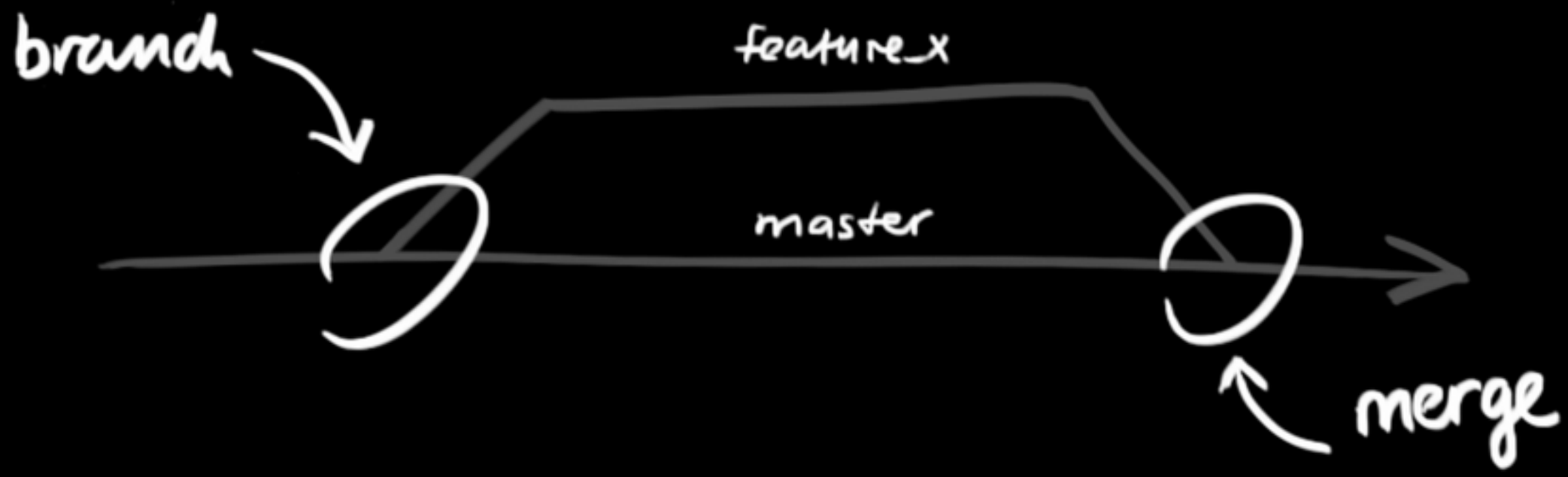
```
git checkout -b feature_x
```

switch back to master

```
git checkout master
```

and delete the branch again

```
git branch -d feature_x
```



update & merge

```
git merge <branch>
```

in both cases git tries to auto-merge changes. Unfortunately, this is not always possible and results in *conflicts*. You are responsible to merge those *conflicts* manually by editing the files shown by git. After changing, you need to mark them as merged with

```
git add <filename>
```

before merging changes, you can also preview them by using

```
git diff <source_branch> <target_branch>
```

http://rogerdudler.github.io/git-guide/

log

in its simplest form, you can study repository history using.. `git log`

You can add a lot of parameters to make the log look like what you want.

To see only the commits of a certain author:

```
git log --author=bob
```

To see a very compressed log where each commit is one line:

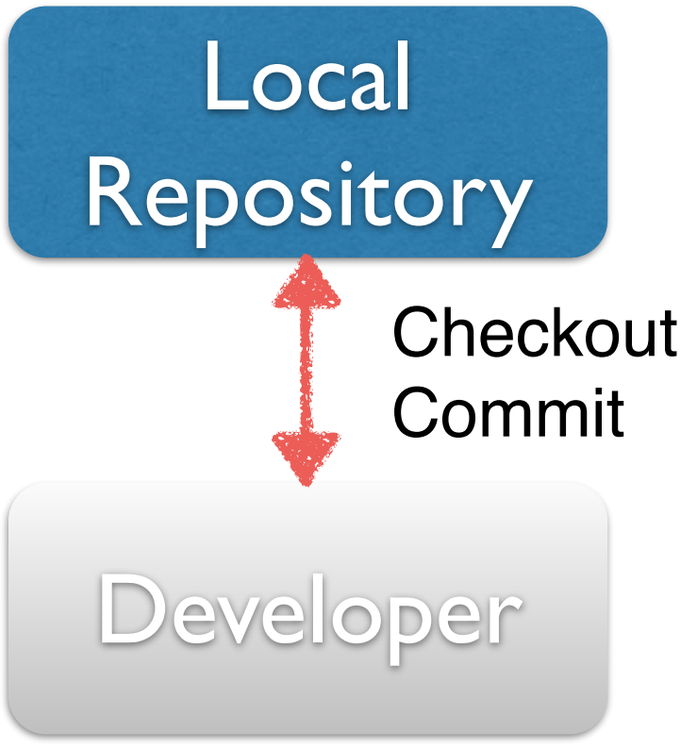
```
git log --pretty=oneline
```

Or maybe you want to see an ASCII art tree of all the branches,

decorated with the names of tags and branches:

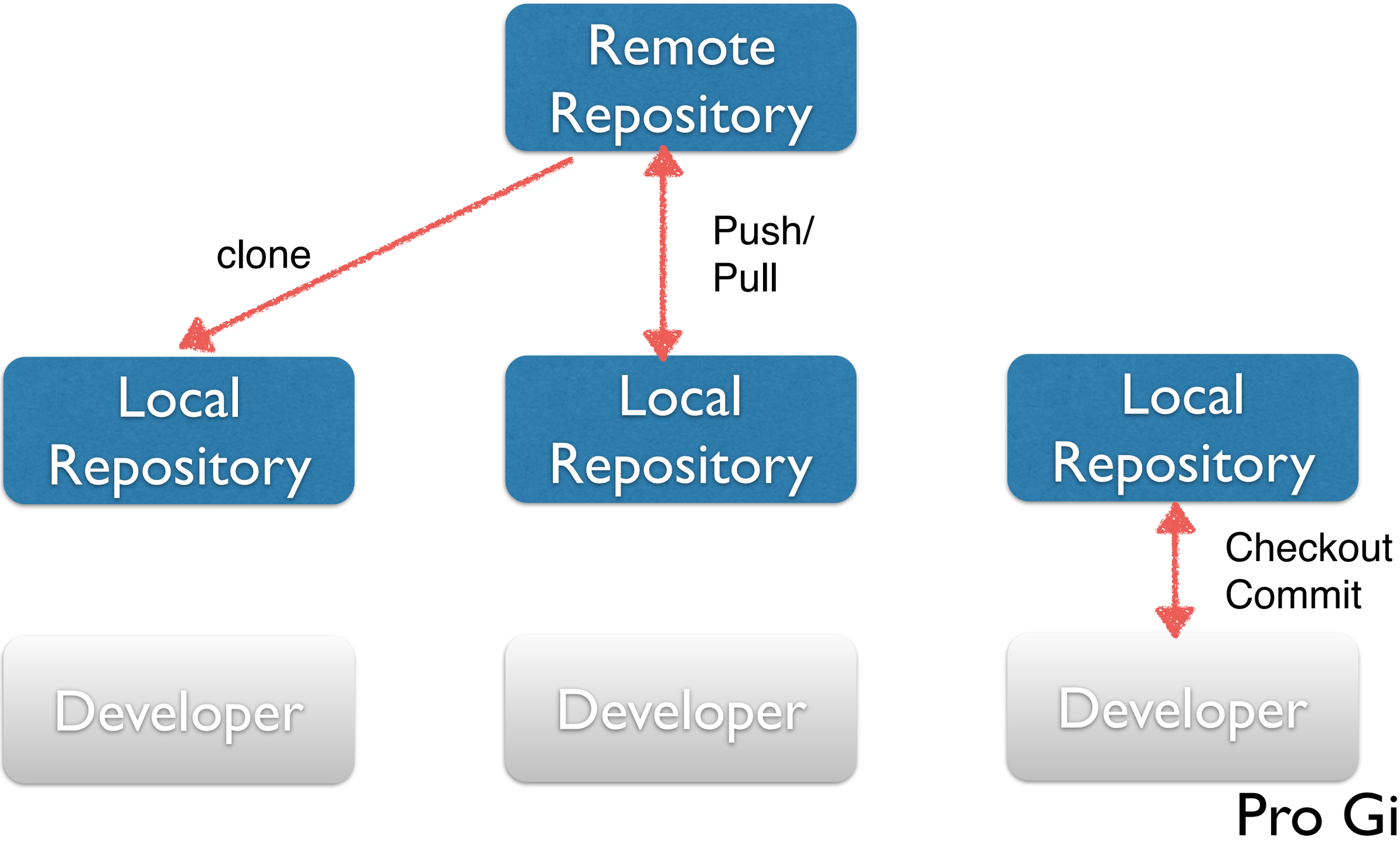
```
git log --graph --oneline --decorate --all
```

Git basic workflow



Pro Git

Git basic workflow



Pro Git

checkout a repository

create a working copy of a local repository by running the command

```
git clone /path/to/repository
```

when using a remote server, your command will be

```
git clone username@host:/path/to/repository
```

pushing changes

Your changes are now in the **HEAD** of your local working copy. To send those changes to your remote repository, execute

```
git push origin master
```

Change *master* to whatever branch you want to push your changes to.

If you have not cloned an existing repository and want to connect your repository to a remote server, you need to add it with

```
git remote add origin <server>
```

Now you are able to push your changes to the selected remote server

branching

remote repository

```
git push origin <branch>
```

update & merge

to update your local repository to the newest commit, execute

```
git pull
```

in your working directory to *fetch* and *merge* remote changes.

http://rogerdudler.github.io/git-guide/

Fix merge conflicts in Git?

http://stackoverflow.com/questions/161813/fix-merge-conflicts-in-git

2617

Is there a good way to explain how to resolve merge conflicts in Git?

git | git-merge | merge-conflict-resolution | git-conflict-resolution

share improve this question

edited Aug 1 '14 at 0:57
Cupcake
45.8k • 14 • 107 • 129

asked Oct 2 '08 at 11:31
Spoike
40.3k • 30 • 108 • 133

asked 6 years ago
viewed 1176114 times
active 1 month ago

1007

8 The following blog post seems to give a very good example on how to handle merge conflict with Git that should get you going in the right direction. Handling and Avoiding Conflicts in Git – mwilliams Oct 2 '08 at 11:40

You can configure a merge tool (kdiff3 jebaird.com/2013/07/08/...) and then use git mergetool. When you're working in large developer teams you'll always encounter merge conflicts. – Grady G Cooper Apr 18 at 5:37

Don't forget that you can mitigate most merge conflicts by regularly merging downstream! – Ant P Jul 27 at 9:50

add a comment

16 Answers

active | oldest | votes

1550

Try: `git mergetool`

It opens a GUI that steps you through each conflict, and you get to choose how to merge. Sometimes it requires a bit of hand editing afterwards, but usually it's enough by itself. It is much better than doing the whole thing by hand certainly.

As per @JoshGlover comment:

The command doesn't necessarily open a GUI unless you install one. Running `git mergetool` for me resulted in `vimdiff` being used. You can install one of the following tools to use it instead: `meld`, `opendiff`, `kdiff3`, `tkdiff`, `xxdiff`, `tortoisemerge`, `gvimdiff`, `diffuse`, `ecmerge`, `p4merge`, `araxis`, `vimdiff`, `emerge`.

share improve this answer

edited Aug 5 at 14:20
kenorb
7,374 • 6 • 56 • 54

answered Oct 2 '08 at 17:50
Peter Burns
22.5k • 6 • 26 • 47

Linked

1 Why do I have (master|MERGING) on the command line and how do I get rid of it?

0 Git Automatic merge failed; fix conflicts and then commit the result.

0 Merge conflict when

https://try.github.io/levels/1/challenges/1

1.1 · Got 15 minutes and want to learn Git?

tryGit

Git allows groups of people to work on the same documents (often code) at the same time, and without stepping on each other's toes. It's a distributed version control system.

Our terminal prompt below is currently in a directory we decided to name "octobox". To initialize a Git repository here, type the following command:

git init

TryGit—987x310

```
Press enter to submit commands

>
```

This repository Search

Pull requests Issues Gist

COMP3111 / questionsJ

Description

Short description of this repository

48 commits

Branch: gh-pages

question

Merge pull request #9 from logchan/gh-page

hunkim authored 4 days ago

css	Created F	5 days ago
js	Created F	5 days ago
test	Added mo	month ago
.gitignore	Upated CSS	5 days ago
CNAME	Changed the domain name	26 days ago

Where should we fork this repository?

@hunkim

@PAMSE

@daumkaka...

@COMP3111

tch 5 Star 6 Fork 21

or Cancel

Code

Issues 6

Pull requests 0

Wiki

Pulse

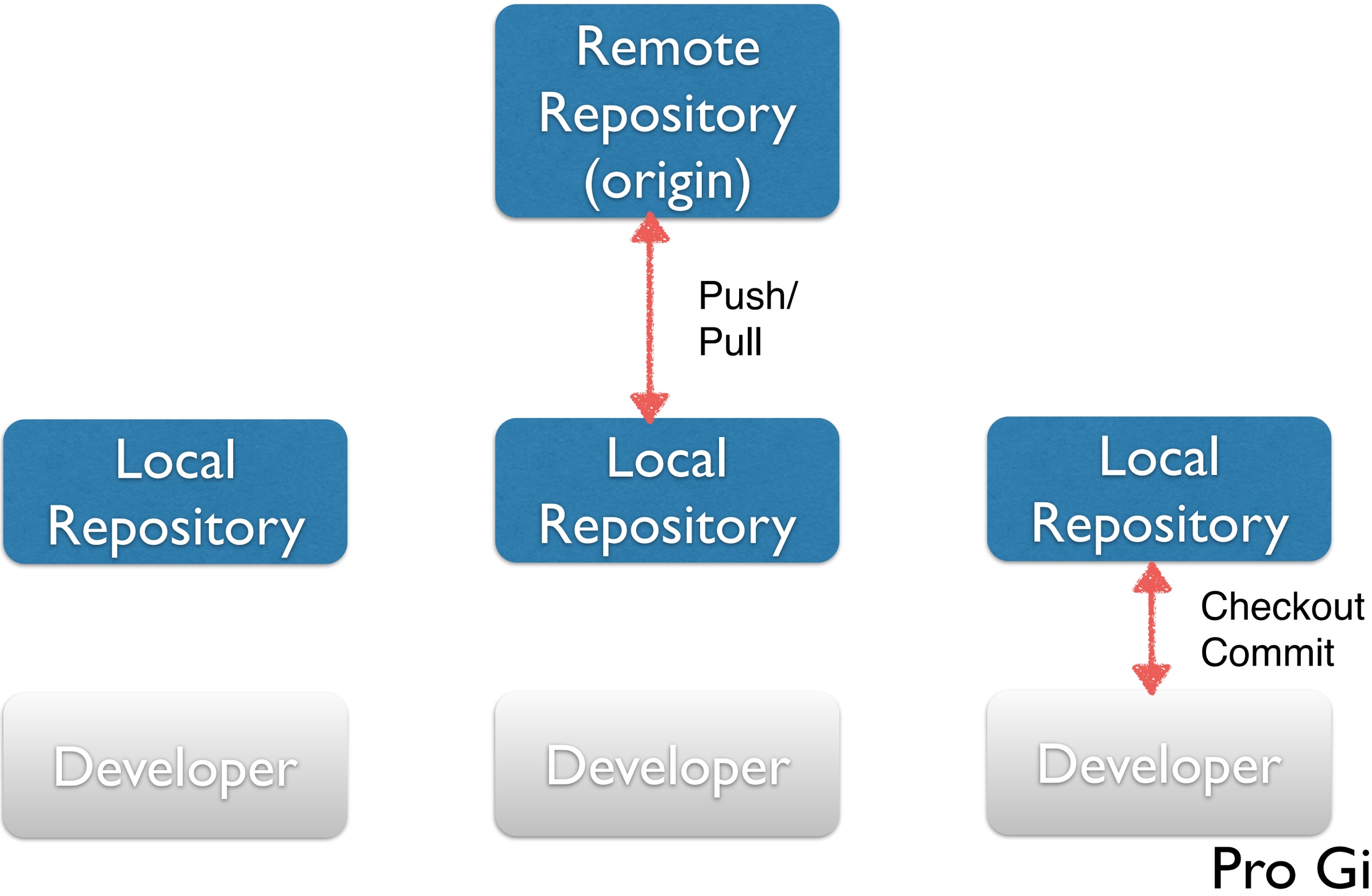
Graphs

Settings

SSH clone URL

git@github.com:CO

Github basic workflow

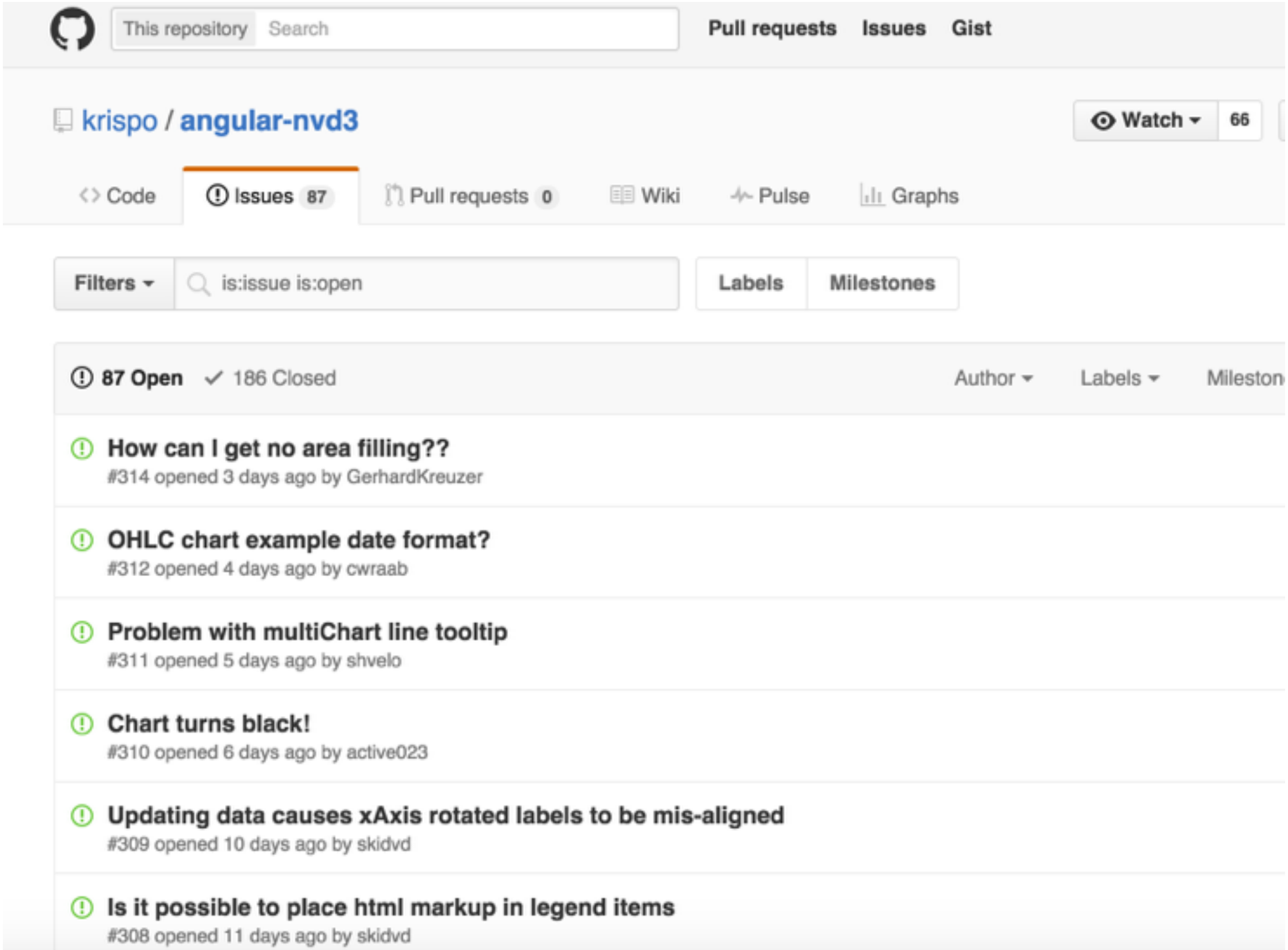


Pro Git

SE Methodologies

- Requirements/Documentation: trello
- Version control system: git
- Issue tracking system: github
- Code review/pair programming: github
- Testing: JUnit
- Continus Integration: Jenkins

Issue tracker



Issue tracker

Wrong Max/Min yAxis on multiChart v1.0.4-dev #286

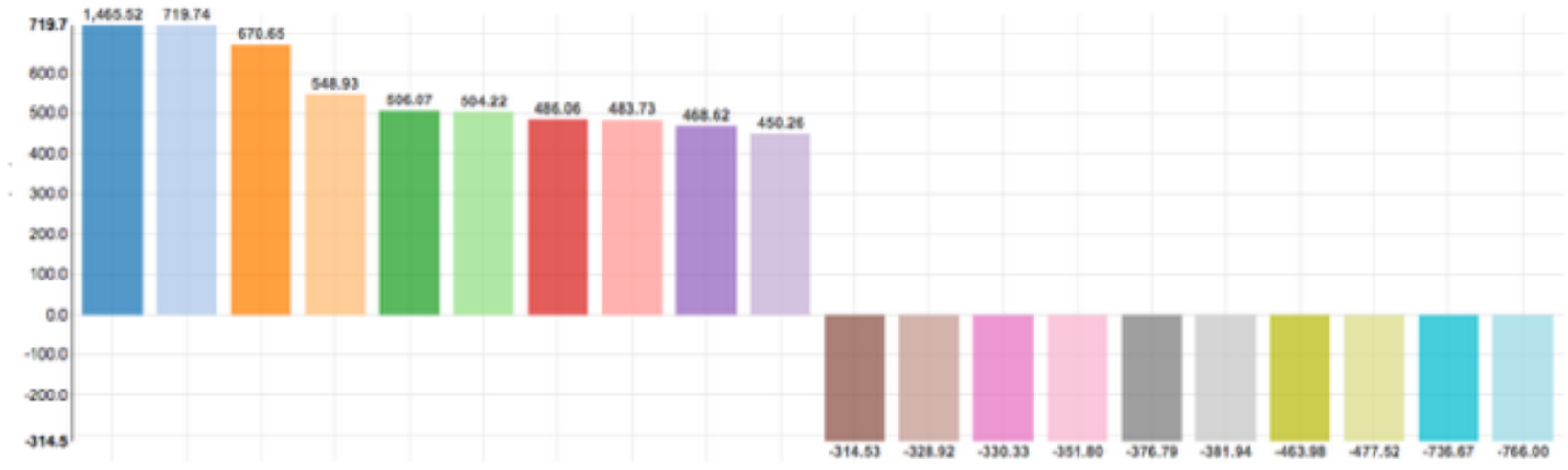
Open **hunkim** opened this issue on Nov 22, 2015 · 9 comments

hunkim commented on Nov 22, 2015

I'm using v1.0.4-dev and update data using:

```
$scope.api.updateWithData(data);
```

However, the yAxis in the graph did not get the right range.



Did I miss something?

Commit log

- What should we write?
- Make sure others can understand what did you do by reading it.
- Can you recall your changes months later.

Commit log (bad)

- empty!
- “bug fix”, “more work”, “minor changes”, “oopsie”, “wtf”

http://www.codelord.net/2015/03/16/bad-commit-messages-hall-of-shame/

Linus Torvalds

```
A good commit message looks like this:

        Header line: explaining the commit in one line

        Body of commit message is a few lines of text, explaining things
        in more detail, possibly giving some background about the issue
        being fixed, etc etc.

        The body of the commit message can be several paragraphs, and
        please do proper word-wrap and keep columns shorter than about
        74 characters or so. That way "git log" will show things
        nicely even when it's indented.

        Reported-by: whoever-reported-it
        Signed-off-by: Your Name <youremail@yourhost.com>

where that header line really should be meaningful, and really should be
just one line.  That header line is what is shown by tools like gitk and
shortlog, and should summarize the change in one readable line of text,
independently of the longer explanation.
```

https://github.com/torvalds/subsurface/blob/master/README#L16

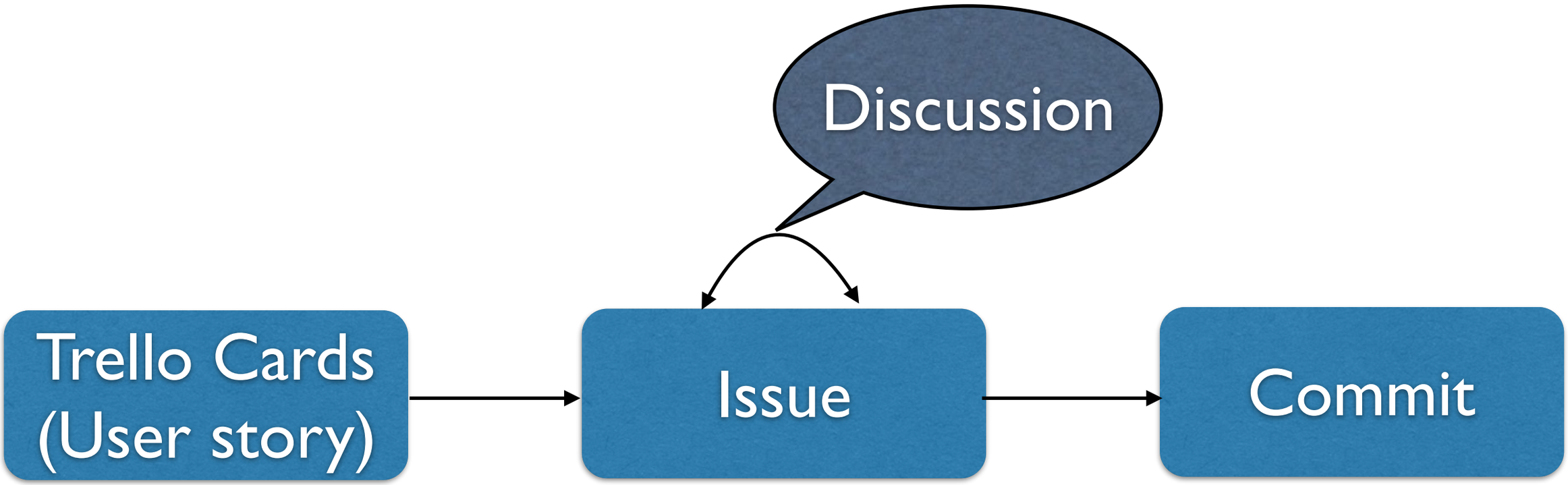
Linus Torvalds

```
A good commit message looks like this:

        Header line: explaining the commit in one line

        Body of commit message is a few lines of text, explaining things
        in more detail, possibly giving some background about the issue
        being fixed, etc etc.

        The body of the commit message can be several paragraphs, and
        please do proper word-wrap and keep columns shorter than about
        74 characters or so. That way "git log" will show things
        nicely even when it's indented.

        Reported-by: w
        Signed-off-by:

where that header line really should be meaningful, and really should be
just one line.  That header line is what is shown by tools like gitk and
shortlog, and should summarize the change in one readable line of text,
independently of the longer explanation.
```

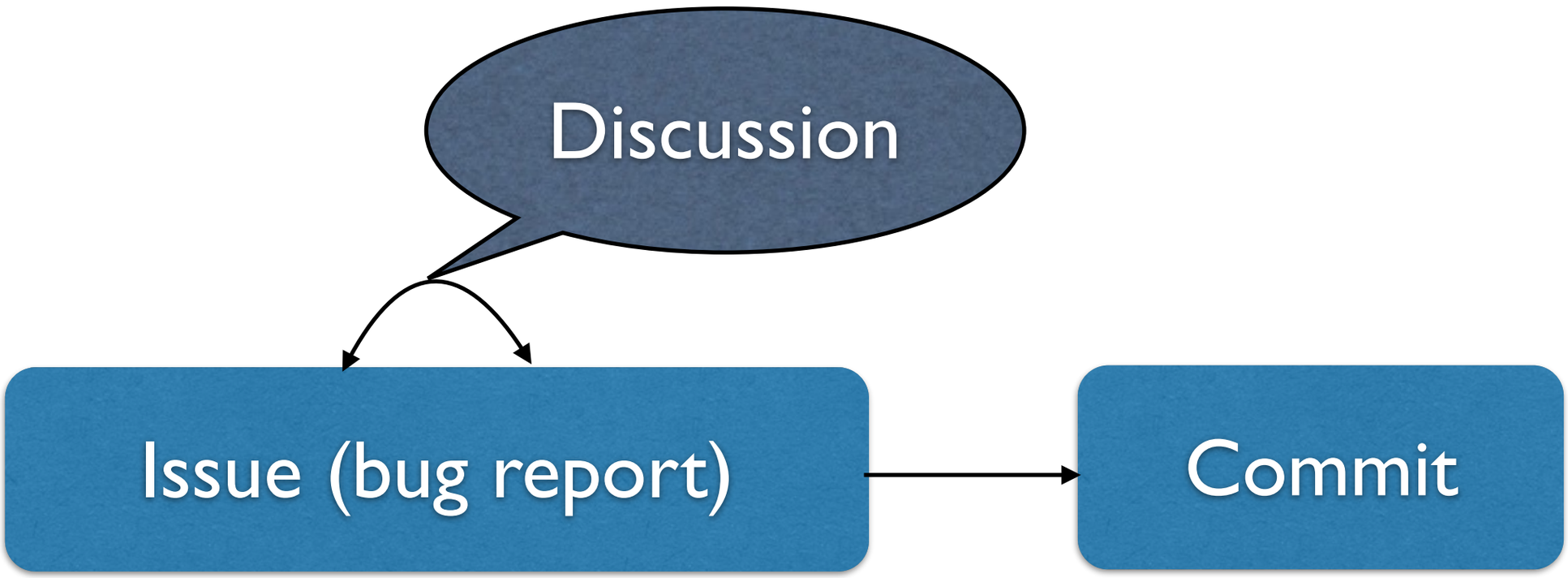
issue #

https://github.com/torvalds/subsurface/blob/master/README#L16

Development process - New features



Development process - Fixes



Summary

- From ideas to products
  - in a reasonable period (a week)
  - with reasonable SE methodologies
- HW/Projects
  - Trello/Issues first
  - Coding/commits later
  - We will read/check them

HW tonight

- Create a trello for your homework/project
- Create github id/repos for your homework/ project

Next
Linux/AWS 101